

# INFOCLUB

REVISTA

DE INFORMATICA SI CALCULATOARE

1  
1991

LAPTOP COMPUTER

FLASH



# CUPRINS CONTENTS ©INFOCLUB 1/91

- 4 Flash:** Vreți să cumpărați un laptop computer?
- 9 Șah:** Șah-computer
- 13 Anchetă:** Informatica în România
- 15 Contribuții:** Proiectul „Elect' 90”
- 16 Actualitatea PC:** Elemente practice de grafică EGA/VGA
- 23 Spot:** Placa Hercules II
- 28 Computer world:** Borland umple ferestrele goale...; IMB lasă friu liber...;
- 30 Laborator Spectrum:** Rutină grafică pentru umplerea contururilor
- 32 Ghidul utilizatorului:** MS-DOS pentru calculatoare personale; Turbo Pascal, versiunile 5.0 și 5.5;
- 40 Computer world:** NeXT din nou în... actualitate
- 43 Atelier Spectrum:** Bright pentru HC-85
- 44 Aplicații pentru toți:** Un instrument de lucru în redacția noastră — calculatorul

Revistă trimestrială  
de informatică  
și calculatoare

ANUL II — NUMĂRUL 2

ADRESA: Piața „Presa liberă” nr.1,  
79781 București

TELEFON: 17 72 44 sau 17 60 10,  
interior: 1151-1258

## Comitetul Director

Ioan ALBESCU  
Gheorghe BADEA  
Mihaela GORODCOV

## Colegiul Științific

Dr. mat. Stelian NICULESCU  
(Ministerul Învățământului și Științei);  
dr. ing. Nicolae TĂPUȘ și  
dr. ing. Valeriu IORGA (Institutul  
Politehnic București, Fac. de  
Automatică); cercet. ing. Eugen  
GEORGESCU și cercet. Ion  
DIAMANDI (Institutul de Tehnică  
de Calcul).

## Culegere pe calculator:

Elena Vasilief; Mariana Badea

Corectură: Mariana Badea

## Prezentare grafică:

Gabi Cătălinoiu

## Administrația:

Editura „Presa Liberă”

## Tiparul:

Întreprinderea „Arta Grafică”

Abonamentele se pot efectua pe adresa redacției prin mandat postal pe numele Badea Gheorghe, urmînd ca abonamentul să fie expediat prin poșta la adresa indicată.

Pentru cititorii din instituții, unități de învățămînt, întreprinderi de stat și particulare numărul minim de abonamente este de 50 exemplare/apariție, putînd beneficia, în acest fel, de o reducere de 20%. Expedierea „abonamentului colectiv” se va face de către redacție prin colet postal la adresa indicată.

**Preț de vânzare: 35 lei**



Avînd sediul în Boston, Massachusetts, INTERNATIONAL DATA GROUP este liderul mondial cu privire la serviciile informaționale și la tehnologia obținerii informației, cu un venit anual de 620 milioane US \$ și 3 800 de angajați.

Divizia dedicată expozițiilor, WORLD EXPO CORPORATION organizează 49 de expoziții și conferințe de calculatoare în 18 țări.

Divizia sa de publicistică și editare, IDG COMMUNICATION publică 150 de ziare și reviste în 50 de țări. Divizia cercetare, INTERNATIONAL DATA CORPORATION (IDC) este liderul mondial al analizei și prospecțiilor de marketing în domeniul calculatoarelor.

INFOCLUB este o publicație a International Data Group (IDG), cel mai mare editor de reviste de informatică și calculatoare din lume. În fiecare lună, 25 de milioane de oameni citesc una sau mai multe publicații IDG.

Publicațiile IDG includ: ARGENTINA: Computerworld Argentina; ASIA: Computerworld Hong Kong, Computerworld Southeast Asia, Computerworld Malaysia, Computerworld Singapore, Infoworld Hong Kong, Infoworld SE Asia; AUSTRALIA: Computerworld Australia, PC World, Macworld, Lotus, Publish; AUSTRIA: Computerwelt Oesterreich; BRASILIA: DataNews, PC Mundo, Automacao and Industria; BULGARIA: Computerworld Bulgaria, Computer Magazine; CANADA: ComputerData, Direct



# ECOURI ȘI... CERTITUDINI

Așadar, iată-ne din nou împreună, la cel de-al doilea număr al revistei INFOCLUB!

Am primit la redacție sute de telefoane și scrisori cu sugestii, observații, critici și chiar... laude și încurajări pe care nu am vrut să le punem în capul listei pentru a nu părea lipsiți de modestie.

Am acordat prea puțin spațiu calculatoarelor Sinclair Spectrum? Se poate! Și putem invoca aici scuza spațiului prea restrâns și a faptului că am dorit să acoperim cât mai multe tipuri de calculatoare.

Am tratat insuficient problema PC-urilor? Perfect adevărat, în mare parte din aceleași motive!

Unii ne-au reproșat chiar spațiul prea restrâns dedicat noutăților din domeniu! Dar, INFOCLUB-ul dorește să fie o revistă orientată spre aplicații practice, lăsând noutățile și articolele generale pe seama... altor publicații.

De aceea, noi, am încercat și vom încerca să acoperim cât mai mult, atât prin intermediul INFOCLUB-ului, cât și al suplimentelor sale dedicate, astfel, încât să reușim să satisfacem un cât mai mare număr de cititori și utilizatori de tehnică de calcul.

Avem posibilitatea să o facem și, chiar, certitudinea că o putem realiza! Și iată de ce!

După cum cred că ați aflat din diferite surse mass-media, INFOCLUB este membru IDG, International Data Group - USA, cel mai important editor de reviste de informatică din lume.

Binecunoscut pe întreg mapamondul, IDG este sinonim, pentru milioane de cititori și utilizatori, cu titluri binecunoscute de publicități începând cu Computer World, fondată în 1967, cu Info World, PcWorld, MacWorld, Network World, Digital News, Federal Computer Week și multe multe altele și terminând cu cele mai recent înființate, mai ales în Europa răsăriteană.

Reviste, ziare, publicații de informație în domeniu, de la jocuri până la aplicații de ultimă oră apar sub antetul IDG în America Latină, Australia, Noua Zeelandă, China, Japonia, Hong Kong, Europa, Statele Unite ale Americii etc. (așa cum reiese și din caseta tehnică).

Informația privită ca o importantă resursă economică și educarea „la zi” a oamenilor cu tot ce se întâmplă într-un domeniu atât de dinamic și de vast cum este informatica, sînt cîteva dintre principiile de bază ale IDG.

Revista INFOCLUB, în calitate de membru al IDG se bucură de numeroase avantaje!

Vehicularea unei documentații bogate și variate din care, începînd cu acest număr,

am selectat pentru dumneavoastră cîteva fragmente referitoare la noua linie NeXT și la unele noutăți grupate sub genericul, binecunoscut deja - „Computerworld”.

De asemenea, posibilitatea de a transmite către IDG așa numitele „contribuții” și realizări deosebite ale specialiștilor noștri în domeniu, firește, al calculatoarelor. De aceea, folosim încă odată acest prilej pentru a invita utilizatorii din țară să ne remită pe adresa redacției astfel de materiale spre a fi transmise în rețeaua de noutăți a IDG. Este o pîrghie importantă cu multiple avantaje și pe care vă invităm să o folosiți, o deschidere ce poate fi benefică pentru ce s-a realizat și se va realiza în continuare în acest domeniu în țara noastră.

Despre IDG se pot scrie foarte multe! O companie „tentaculară”, cu trăsături aparte în peisajul general al marilor grupuri, care pune un accent deosebit pe educarea și școlarizarea personalului.

„Cred că noi investim în instruirea personalului mai mult decît alte companii. Este important pentru succesul nostru. Am ajuns la concluzia că cele mai bune rezultate ale unei investiții vin din școlarizare și educație”, a spus, într-un interviu, președintele și fondatorul IDG, domnul Patrick McGovern.

Aceasta, alături de contactele și colaborările companiei cu firme de primă mărime și importanță în tehnica de calcul mondială, constituie cheia succesului IDG.

În ceea ce ne privește, INFOCLUB rămîne o revistă deschisă tuturor sugestiilor și dialogului cu utilizatorii și cititorii care încearcă să se adapteze în funcție de dinamica domeniului.

Cam atât deocamdată. Și vă așteptăm în continuare!

*Michelle Gunders* *Sh. Bodie*

Access, Graduate CW, Macworld; CHILE: Informatica, Computacion Personal; COLUMBIA: Computerworld Columbia; CEHOSLOVACIA: Computerworld Czechoslovakia, PC World; DANEMARCA: CAD/CAM WORLD, Computerworld Denmark, Communication World, PC World, Macworld, Unix World, PC LAN World; FINLANDA: Mikro PC, Tietovikko, Tietotekniikka; FRANȚA: Le Monde Informatique, Distributique, InfoPC, Telecoms International; GERMANIA: Computerwoche, Information Management, Amigawelt, PC Woche, PC Welt, Unix Welt, Macwelt RD; GRECIA: Computerworld, PC World, Macworld, Infoworld; UNGARIA: Computerworld SZT, Mikrovilag; INDIA: Computers and Communications; ISRAEL: People and Computers; ITALIA:

Computerworld Italia, PC World Italia; JAPONIA: Computerworld Japan, Macworld; COREEA: Computerworld, PC World; MEXIC: Computerworld Mexico, PC Journal; OLANDA: Computerworld Netherland, PC World, Amiga World; NOUA ZEELANDĂ: Computerworld New Zealand, PC World New Zealand; NIGERIA: PC World Africa; NORVEGIA: Computerworld Norge, PC World Norge CAD/CAM, Macworld Norge; CHINA: China Computerworld, China Computerworld Monthly; FILIPINE: Computerworld Philippines, PC Digest/PC World; POLONIA: Computers Magazine, Computerworld; ROMÂNIA: Infoclub; SPANIA: CIM World, Comunicaciones World, Computerworld Espana, PC World, Amiga World; SUECIA: ComputerSweden, PC/Nyhetera, Mik-

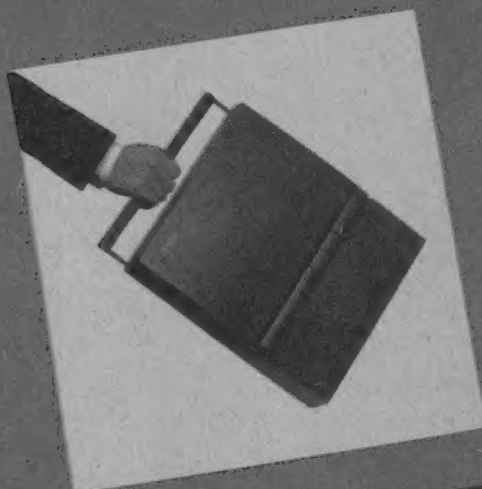
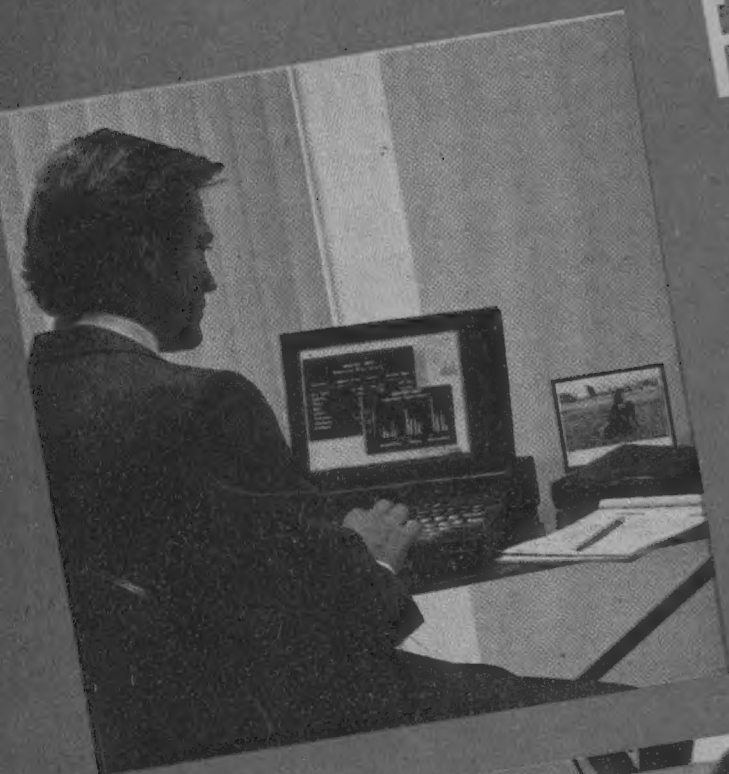
rodatorn, PC World, Macworld; ELVEȚIA: Computerworld Switzerland, Macworld; TAIWAN: Computerworld Taiwan, PC World, Publish; TAILANDA: Computerworld; TURCIA: Computerworld Monitor, PC World/Turkiye; MAREA BRITANIE: Graduate Computerworld, PC Business World, ICL Today, Lotus UK, Macworld U.K.; STATELE UNITE: Amiga World, A+, CIO, Computerworld, Digital News, Federal Computer Week, GamePro, IDG Books, InfoWorld, Macworld, NextWorld, Network World, PC Games, PC World, Portable Office, PC Letter, Publish, Run, Sun Tech Journal; URSS: MIR PC, Computerworld URSS, Network, Manager Magazine; VENEZUELA: Computerworld Venezuela, Micro Computerworld; YUGOSLAVIA: Moj Mikro.

# laptop

COMPUTER

FLASH

micro



# VREȚI SĂ CUMPĂRAȚI UN LAPTOP-COMPUTER? FOARTE BINE, DAR...

Utilizarea în masă a calculatoarelor personale compatibile modelelor XT sau AT a implicat pentru producători un foarte incitant pariu: realizarea unui PC care să poată fi luat în călătorie și să servească drept bază mobilă de calcul. Acest computer trebuia să poată fi utilizat practic oriunde, așezat pe cele mai diferite suprafețe, cel mai adesea ținut chiar... pe genunchi. Un computer complet, ținut și operat pe genunchi! Numele se născuse: laptop computer. Mai trebuia să fie și produs!

Prima problemă care trebuia rezolvată era aceea a contradicțiilor care există între diferitele module constructive, deja standardizate în tehnica de calcul. Procesoarele din seria 8086, 80286 și 80386, floppy discurile de 3,5 inch, discuri hard, afișaje contrastante, dar economice în consum, tastatura cât mai ergonomică și insensibilă la contaminanți, toate acestea trebuiau incluse într-o carcasă care să ofere bune posibilități de răcire, să asigure ecranarea radiației electromagnetice, precum și stabilitate structurală. În carcasă mai trebuia făcut loc și unei surse de alimentare care să fie simultan rezistentă climatic, ușoară, etanșă, să aibă o viață cât mai lungă, să asigure densități de energie mari pe volum și să poată fi încărcată rapid.

Tot acest ansamblu trebuie să reziste la proba uzuală de cădere de la înălțimea unui birou pe o pardoseală de ciment, indiferent de partea care vine prima în contact cu solul. În final, produsul trebuie să asigure compatibilitate software cu modelele de PC compatibile IBM.

Simpla înșiruire a cerințelor care au stat în fața proiectanților indică obstacolele formidabile care au trebuit depășite pentru realizarea acestor bijuterii tehnice. Și aici, ca în orice alt domeniu, în care ingineria miniaturizării a depășit limitele aparente ale posibilității, primii producători in-

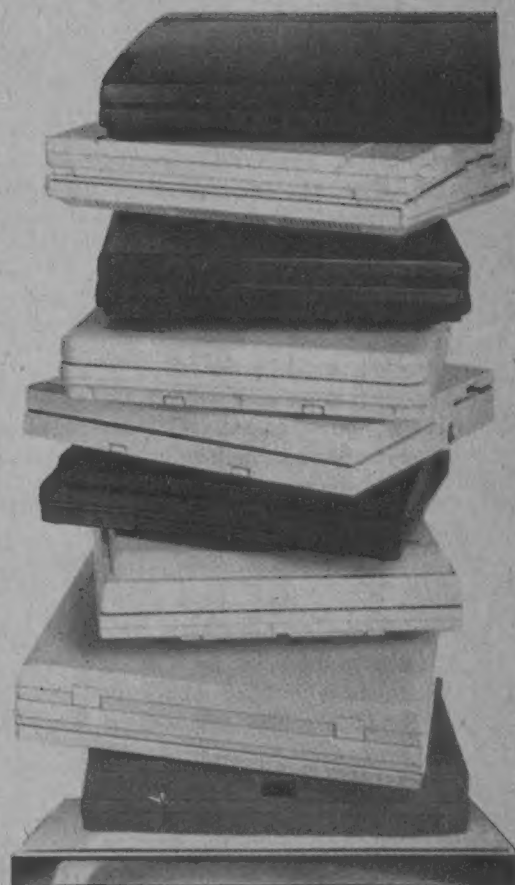
dustriali de computere laptop au fost japonezii cu legendara serie **Toshiba T 1000, T 1200**, etc. Această serie de succes comercial a evoluat ajungând astăzi până la modelele **T 5100** și **T 5200** cu discuri hard de 100 și 200 MBytes.

## UN LAPTOP ÎNSEAMNĂ...

**Procesoare.** Cu predilecție sînt folosite procesoarele V20 (compatibil 8086) sau cele realizate în tehnologia CMOS: 80C86 și 80C286. Cu toate acestea se întîlnesc și modele „sport” care demonstrează solidă performanță a unui 386SX cu 16 MBytes de memorie operativă.

**Afișajul.** Computerele de tip laptop sînt obligate să folosească tipuri de afișaje caracterizate prin consumuri reduse: cristale lichide, plasmă sau chiar tuburi plate color, avînd tunul electronic așezat lateral, așa cum este cazul modelului Hitachi HL 500C, unde rezoluția este 640 x 480 cu 8 culori!

Standardele implementate sînt, în general, orientate în două direcții principale. În primul rînd, pentru afișaje ieftine cu cristale lichide se folosesc formate alfanumerice și grafice nepretențioase, care nu respectă nici un standard grafic cunoscut. Pentru mode-





lele mai evolute se folosesc afișaje compatibile EGA și VGA monocrom și, rar, color.

Pentru asigurarea consumurilor reduse, ecranele sînt stinse după o perioadă de inactivitate a operatorului; prima apăsare pe taste va readuce conținutul ecranului înapoi.

### Unitățile de floppy-disc.

În calculatoarele de tip laptop s-a standardizat existența unui floppy de 3,5 inch, cu o capacitate de 1,44 MBytes. În tipurile cu consum ultra-redus s-au introdus RAM-Cards, plăci pe care sînt montate memorii RAM de 32, 64 sau 128 KBytes și care pot fi conectate la computer prin intermediul unui conector adecvat. Aceste RAM-Cards, deși rezolvă parțial nevoia unei memorii mai mari, au o serie de dezavantaje: nu toate pot reține datele și nu sînt standardizate.

Unitățile cunoscute de 5,25 inch, pot fi folosite ca echipamente externe cu sursă proprie de alimentare și cu cabluri de conectare externe. Fără îndoială, acest tip de utilizare reduce mult caracteristica de portabilitate, dar asigură compatibilitatea cu suportul de informații, cerință primordială în multe aplicații.

### Unitățile de hard-discuri.

Pe lângă cunoscutele unități de 3,5 inch, în laptops au apărut și unități de 2,5 inch. Capacitățile oferite nu sînt deloc modeste, începînd cu 40 MBytes pînă la discuri „serioase” de 100 MBytes care pot memora între 20 000 și 50 000 pagini A4 dactilografiate. Volumul și greutatea acestor pagini memorate depășesc cu mult pe cele ale computerului. Totodată, pentru aplicații pretențioase sînt livrabile și unități externe de discuri de 100 MBytes, cu surse proprii de alimentare.

Tot ca unități externe sînt livrabile și unități de casetă magnetică tip streamer, extrem de utile în salvări sau arhivări ale conținutului unor hard discuri.

### Memoria internă.

Dacă nimic nu ne mai surprinde în tehnologia actuală,

atunci capacitatea memoriilor interne apare ca un argument uzual: 1 MByte, extensibil la 4 sau chiar la 16 MBytes! Programele cele mai exigente și „infometate” de memorie încap acum în „colegul” nostru electronic cu ajutorul căruia, în drumul de la București la Brașov — atît durează acumulatorii — putem să proiectăm, simulăm, corectăm și trasăm placa noului circuit electronic pe care îl oferim spre integrare în noul și ultra-modernul camion...

### Comunicații.

Modelele laptop își dovedesc utilitatea mai ales în cazurile în care posesorul are o activitate care reclamă mobilitate. În mod normal, în fiecare laptop întîlnim o interfață serială, multe dintre ele posedînd și un modem încorporat care lucrează la vitezele uzual întîlnite, deci pînă la 2400 Bauds. La fel de des întîlnit este și cazul existenței unei cuple V.24 la care se poate atașa un modem acustic portabil livrat cu tot cu cupelele de izolare acustică care se montează pe receptorul telefonic. Mai nou, producătorii oferă și plăci de tip telefax la un pachet soft adecvat.

Un exemplu este Toshiba Microlink 2400XL care comunică destul de modern: V.21 duplex la 300 bit/s, V.22 cu 1200 și 2400 bit/s duplex. În plus, placa mai lucrează și în mod Fax folosind CCITT V.27 semiduplex, la viteze de 2400 sau 4800 bit/s.

Oricum ar fi, placa de comunicații, modemul precum și pachetul soft trebuie alese după o consultare cu poșta națională pentru a evita orice surprize. Trebuie remarcat că nu orice laptop sud-est-asiatic „ieftin” este capabil să se înțeleagă fără probleme cu un Felix-PC autohton sau cu importuri „standard PC”. Situația se agravează dacă vrem să comunicăm cu mașini de la nivelul unui minicalculator în sus.

Aceeași situație se întîlnește și în cazul aplicațiilor de tip Fax. Și aici, regula de aur: „întreabă de o mie de ori și cumpără o dată!” se dovedește a fi deosebit de utilă. Mai ales că un laptop nu este tocmai ieftin...

### Greutatea, dimensiunile și, mai ales, costul!

În general, computerele laptop nu sînt tocmai ușoare. Trebuie să ne așteptăm la greutatea cuprinse între 3 și 6 kg, în funcție, mai ales, de tipul acumulatorilor folosiți.

Costurile sînt în general mai ridicate decît ale calculatoarelor personale de birou. De exemplu, un PC uzual, tip AT, poate costa pe piața germană în jur de 2500 DM, dar orice încercare de apropiere de un laptop model AT ne va face să numărăm la casă peste 5000 DM! Fără îndoială și aici performanța se plătește. Și încă scump... Un calcul simplu arată că 1 kg laptop costa între 800 și 2000 DM!

### Sistemele de alimentare.

Pentru a putea asigura alimentarea unui laptop trebuie să se facă o alegere deosebit de dificilă, mai ales din punctul de vedere al producătorului. Fără îndoială, produsul trebuie să fie: rapid, cu memorie operativă mare, cu ecran vizibil și în lumină directă, cu memorie mare pe discuri, dar, totodată, ușor, independent de surse de energie, insensibil la temperatură, umiditate, șocuri.

Toate aceste cerințe primare creează un tablou contradictoriu, rareori rezolvat satisfăcător. Vom întîlni astfel modele laptop cu consum redus, la care bateria sustine sistemul peste 60 ore, dar cu procesoare arhaice — Z80 — sau modele ultraputernice ale căror acumulatori ajung doar pentru circa 2 ore. Totodată, tipul acumulatorilor variază foarte mult, de la grelele baterii cu plumb-acid pînă la periculoasele baterii cu litiu. Practic, deși un laptop este portabil, mai bine folosim adaptorul și îl alimentăm la rețea!

Acest tablou relativ sumbru trebuie totuși completat cu faptul că utilizatorul trebuie să acorde o atenție deosebită acumulatorilor: o scurgere minoră de electrolit poate distruge iremediabil computerul. La multe modele, o funcție software asigură indicarea gradului de încărcare a acumulatorilor, iar sub o anu-

mită limită, computerul avertizează sonor și vizual necesitatea încărcării acestora înainte ca să fie distruși prin descărcare excesivă.

**Software.** Dacă PC-laptop pe care îl achiziționați conține un procesor Intel care depășește nivelul 8086 precum și un hard disc, atunci veți avea acces la sisteme de operare evoluat: DOS 3.xx sau DOS 4.01. Prin intermediul acestora puteți asigura rularea unui tezaur de pachete soft care acoperă, practic, toate aplicațiile uzuale. Fără îndoială, trebuie verificată utilizarea oricărui pachet înainte de generalizare în sistemul companiei în care lucrați.

A doua posibilitate constă în a avea un laptop care are un sistem de operare propriu. În acest caz este uzual să fie incluse un procesor de texte, o bază de date, un program de calcul cu tabele tip Lotus, un interpretor Basic și un modul de comunicații. Pe lângă acestea, mai pot fi întâlnite și pachete de grafică comercială sau de gestiune a unui fișier de adrese, cu posibilitatea de a folosi modemul încorporat direct pentru formarea numărului de telefon. În acest al doilea caz, este uzuală funcția de calculator științific și comercial analog modelelor produse de Hewlett-Packard, fie de Texas Instruments.

În ambele variante, ținând cont de caracterul de portabilitate al mașinii, puteți să obțineți și pachete care oferă o interfață comodă între utilizator și sistemul de operare instalat. Asemenea module, numite „Shell” sau „Personal application manager” consumă spațiu în memoria internă și pe discul hard dar scutesc utilizatorul de manevre de prestidigitator pe o tastatură densă și cu asignări multiple.

## **SURPRIZĂ SAU NOUA VIAȚĂ A UNUI VETERAN.**

Apărent, în lumea PC, nu mai există procesoare decât de la 80286 în sus. Totuși, firma Rhisc — bun exemplu și pen-

tru producătorii noștri — oferă un laptop de dimensiunile unui pagini A4, cântărind sub 2 kg, are 5 cm înălțime, bateriile îl susțin 60 ore, RAM-card CMOS și display LCD de 8 linii și 80 caractere.

Ce procesor este folosit? Veteranul Z80 care, la 3 MHz s-a dovedit capabil să rezolve la preț redus — tot computerul costa sub 1 000 DM, în varianta de bază — toate problemele legate de gestionarea periferiei și de pachetele software integrate în produs.

## **Un exemplu recent.**

Un exemplu de laptop tipic îl constituie **Dell 316LT**, un laptop care folosește un procesor 386SX, la 16 MHz, cu o memorie internă de 1 MByte — modelul de bază — un hard-disc de 40 MBytes și un floppy de 3,5 inch. Afișajul este asigurat cu cristale lichide supertwist și asigură confortul standardului VGA: 640 x 480 puncte. În model mai găsim și o interfață serială și una paralelă, precum și un slot de 8 biti pentru extensii tip XT, cum ar fi: rețele locale, fax sau modem. Plătind peste 8 000 DM, puteți cumpăra acest „compumonster” cântărind 7,5 kg și cu dimensiunile 33 x 36 x 8,5 cm.

## **Mic îndrumar de prețuri.**

Fără îndoială, estimarea utilității unui produs se face considerând și prețul pe care îl plătim pentru serviciile sale. Pentru aceasta v-am putea oferi un tabel de prețuri practicate pe piața europeană cea mai apropiată de noi(!): piața germană, anul 1990. Prețurile fiind exprimate în DM, iar caracterul lor pur informativ, pot exista și prețuri mai mari sau chiar mai mici, mai ales în cazul produselor zise de „export”.

Cum sub acest eufemism se ascund produse care nu au trecut normele de atestare tehnică pentru piața respectivă, amînăm intenția noastră pentru altă dată, sau vă putem satisface curiozitatea numai ca urmare a solicitărilor dumneavoastră.

## **Computerworld**

• Poliția din Nevada și-a îmbunătățit de curind rețeaua de telecomunicații care permite ofițerilor și patrulilor de pe stradă să se conecteze la bazele de date locale, federale și de stat în circa 10 secunde! Cu o singură cerere ofițerilor de pe teren, aflați în orice punct al statului pot avea acces foarte rapid la un sistem specializat (LEMS = Lan Enforcement Message Switching system) la informațiile cele mai diverse privind infractorii, criminalii, infracțiunile etc., pentru elucidarea în timp foarte scurt a cazurilor și anchetelor. Înainte de implementarea acestui sistem, utilizatorii trebuiau să acceseze 17 baze de date din diferite puncte ale statului pentru aflarea informațiilor dorite.

LEMS la în considerare centrele de informație și le „îndrumă” la sursele de date adecvate mărind mult viteza întregului proces precum și volumul tranzacțiilor (circa 200 000 pe zi) care se pot face într-o unitate dată de timp, deoarece „viteza și flexibilitatea sînt esențiale pentru prinderea unui criminal”. (Network World — 31 decembrie 1990/7 ianuarie 1991).

• În noiembrie trecut Motorola a anunțat lansarea și producerea puternicului microprocesor 68040, despre care Apple a afirmat că va fi „inima” viitorului Macintosh. Avînd ceasul de 25 MHz, microprocesorul echi-pează deja și stațiile de lucru Hewlett-Packard și NeXT, asigurîndu-le o viteză de lucru de 20 MIPS; M 68040 este de cîteva ori mai rapid decît predecesorul său M 68030 de 40 MHz care echi-pează Macintosh II fx. (Macworld — februarie 1991).

## **Computerworld**



**şah**



**computer**





Vă mai amintiți de ASTRO—64? Vă mai amintiți de acele pasionante articole realizate de U. Văluoreanu, în paginile revistei „Magazin”, cu privire la evoluția programelor de șah pe calculator? În acei ani șahul pe calculator își făcea debutul în țara noastră, eram printre puținele țări care aveau realizate asemenea programe. În ultimii ani, însă, din motive mai mult sau mai puțin obiective, șahul programat din țara noastră a cunoscut o vizibilă stagnare, în timp ce, în mai multe țări din lume această disciplină a informaticii s-a dezvoltat mult, ducând la apariția de numeroase programe de șah, care participă la campionate naționale și la frecvente întreceri internaționale.

## APEL CĂTRE INFORMATICIENI

Prin acest material, care va apărea în mai multe numere ale revistei „INFOCLUB”, intenționăm să inițiem o interesantă acțiune, ca toți cei care se simt atrași de subiectul șahului programat (informaticieni, studenți, elevi) care au posibilitatea să lucreze pe un calculator personal, să încerce să-și realizeze propriile programe de șah.

În această idee, vom prezenta un mini-curs de programare a șahului, sub forma mai multor articole ce vor apărea în această revistă. Vă rugăm să popularizați inițiativa noastră printre prietenii voștri informaticieni!

Cursul nostru, deși teoretic, sperăm să constituie un bun ghid pentru înțelegerea problematicei șahului programat, care ar putea să-i facă pe începătorii în acest domeniu să evite greșelile de concepție, care ar reduce, din start, performanțele programelor lor. De bună seamă, fiecare autor e liber să-și aleagă propriile căi de abordare a acestui inepuizabil subiect, cursul nostru însemnând, de fapt, o colecție de sugestii.

Dacă în urma acestei inițiative vor fi scrise mai multe programe de șah, intenționăm să organizăm cu ele campionate locale sau naționale. De asemenea, cele mai bune dintre aceste programe vor fi desemnate să participe la campionate (sau meciuri) internaționale.

Curaj și succes tuturor acelor care se decid să abordeze acest frumos subiect, al programelor de șah pe calculator! Puteți să ne scrieți pe adresa revistei cu mențiunea „ŞAH—COMPUTER”.

## Sfaturi de început.

• Dacă lucrați pe microcalculatoarele pe 8 biți (CUB—Z, M—118, SINCLAIR etc.) se pare că nu poate fi evitat limbajul de programare ASSEMBLER, dacă se urmărește performanță

de timp, căci, chiar și limbajele C și PASCAL pe aceste microcalculatoare generează cod inefficient. Eventual, se poate scrie în aceste limbaje cu scopul de a depăna programele, ca apoi ele să fie trecute pe calculatoare mai puternice.

• Pe microcalculatoarele pe 16 sau 32 de biți însă (cum sînt cele compatibile IBM—PC), compilatoarele C și PASCAL generează un cod deosebit de eficient și nu se mai justifică nici o reticență în privința eficienței acestor limbaje. E drept, programînd în ASSEMBLER s-ar putea să se mai optimizeze cite ceva, nesemnificativ totuși. Este o iluzie că programarea în limbajul de ansamblare ar aduce avantaje, de fapt, asta face să pierdeți mult timp cu implementarea oricărei idei, și veți avea de depanat multe erori.

• În plus, programînd în limbajele C sau PASCAL, veți putea beneficia de posibilitatea apelului recursiv al procedurilor, lucru neoferit de FORTRAN, COBOL sau ASSEMBLER.

• Evitați, pe cît posibil, folosirea numerelor reale într-un program de șah, căci operațiile cu numere reale consumă mult timp. De fapt, numerele reale pot fi evitate aproape integral într-un program de șah.

• Pentru codificarea tablei de șah, a numerelor pătratelor, a indicatorilor, puteți folosi variabile pe un octet. Totuși, pentru exprimarea cît mai fină a tuturor valorilor oferite de diferite criterii de apreciere a **ponderilor statice** și a **valorilor mini-max**, e bine ca aceste numere să fie întregi, pe 16 biți.

## MODEL INFORMATIC AL JOCULUI DE ŞAH

Aşadar, cu ce începem scrierea unui program de şah?

De bună seamă, primul lucru ce se cere lămurit, este modul de reprezentare în memoria calculatorului a tablei de şah și a pozițiilor pieselor pe tabla de şah.

## Alegerea modelului pentru tabla de şah.

Pare banala aceasta problemă, căci ce poate fi mai simplu decît o matrice TABLA (8,8), ale cărei elemente sînt numere întregi, ce codifică diferite tipuri de piese?

Totuși, cînd scriem un program de şah, tabla în forma ei cea mai simplă de matrice 8 x 8 are inconvenientul că, în mutarea succesivă a unei piese dintr-un pătrat în altul, trebuie făcute 4 teste de fiecare dată, dacă acea piesă n-a depășit marginea tablei, în sus, în jos, în stînga și în dreapta. În plus, accesul la orice element al unei matrice cu două dimensiuni TABLA (i,j) ascunde și o înmulțire  $i \times 8 + j$ .

	6E	6F	70	71	72	73	74	75	76	77
	64	65	66	67	68	69	6A	6B	6C	6D
8	5A	5B	5C	5D	5E	5F	60	61	62	63
7	50	51	52	53	54	55	56	57	58	59
6	46	47	48	49	4A	4B	4C	4D	4E	4F
5	3C	3D	3E	3F	40	41	42	43	44	45
4	32	33	34	35	36	37	38	39	3A	3B
3	28	29	2A	2B	2C	2D	2E	2F	30	31
2	1E	1F	20	21	22	23	24	25	26	27
1	14	15	16	17	18	19	1A	1B	1C	1D
	A	B	C	D	E	F	10	11	12	13
	0	1	2	3	4	5	6	7	8	9
	A	B	C	D	E	F	G	H		

fig.1 Tabla alindrică. Numerotarea în (hexa) a pătratelor ei. Tabla de șah propriu-zisă este în chenarul interior.

Acest motiv face ca să se recurgă la reprezentări ale tablei de șah mai convenabile, care să rezolve anumite probleme. Astfel, cel mai des se folosește așa-zisa **tabla cilindrică**, care se poate vedea în figura 1, care este un vector TABLA (120) conținând 120 de elemente, ce se consideră împărțită pe 12 linii a câte 10 elemente, și care conține liniile și coloanele tablei de șah 8 x 8, mărginite sus și jos cu fișii a câte două linii cu valori negative, iar în stînga și dreapta cu încă o fișie cu valori negative, folosite pentru detecția marginii. Dacă se îndoaie marginea din stînga a tablei, ca să se unească cu marginea din dreapta, se obține o fișie cu două rînduri de pătrate cu valori negative, precum și aspectul de tablă cilindrică, de unde și denumirea. Fișiile de 2 rînduri din margine sînt necesare din cauza mutărilor calului.

Fișiile de margine conțin coduri negative (mod de detecție a marginii tablei), iar celelalte elemente ale vectorului TABLA (120) conțin fie valori zero (pătrate libere), fie valori strict pozitive (codurile pieselor albe sau negre).

	-1	-1	-1	-1	-1	-1	-1	-1	-1
	-1	-1	-1	-1	-1	-1	-1	-1	-1
8	-1		11						-2
7	-1			7			1	7	-1
6	-1					1		6	-1
5	-1	7		4				3	-1
4	-1	1		9		2		7	-1
3	-1	7		1	12		7	1	-1
2	-1	9			7		2		-1
1	-1		9		5	4			-1
	-1	-1	-1	-1	-1	-1	-1	-1	-1
	-1	-1	-1	-1	-1	-1	-1	-1	-1
	A	B	C	D	E	F	G	H	

fig. 2 Codificarea unei poziții de pe tabla de șah.  
Mat în 2 mutări.

### Codificarea pozițiilor pe tabla de șah.

Ce coduri se atribuie diferitelor tipuri de piese? Nu are prea mare importanță acest lucru, totuși, fiecare autor de programe de șah își alege astfel codificarea pieselor, încît să-și rezolve anumite probleme curente: folosirea codului piesei ca index în diferite tabele, aprecierea importanței piesei conform codului ei etc.

În programul ASTRO-64 (realizat pe calculatorul FELIX C-512), de asemenea în LABIRINT-64 și ATOM-64 (realizate pe microcalculatoarele M-118, CUB-Z), s-a ales următoarea codificare a pieselor:

- |                 |                      |
|-----------------|----------------------|
| 1 = pion alb    | 7 = pion negru       |
| 2 = turn alb    | 8 = turn negru       |
| 3 = cal alb     | 9 = cal negru        |
| 4 = nebul alb   | 10 = nebul negru     |
| 5 = regina albă | 11 = regina de negru |
| 6 = rege alb    | 12 = regele negru    |

Încercați să reconstituiți poziția de pe tabla de șah codificată în figura 2. În această figură se vede un cod „-2” după pătratul H8. Este o mică „invenție”, care permite detectarea sfîrșitului tablei de șah cînd aceasta este parcursă secvențial.

### Codificarea mutărilor.

O mutare este o înregistrare cu mai multe cîmpuri: pătratul sursă, pătratul destinație, indicator de mutare specială, ponderea (nota) asociată mutării etc.

Pătratele se codifică ca un singur întreg, pe un octet (sau pe doi octeți pe calculatoare pe 16 biți), conform numerotării din figura 1, de exemplu.

### Generatorul de mutări.

Pentru a putea scrie un subprogram care să genereze lista tuturor mutărilor ce se pot efectua, într-o semimutare, dintr-o poziție dată pe tabla de șah, este necesar să fie precizate următoarele informații:

- conținutul vectorului TABLA (120);
- indicatorul care să spună dacă albul sau negrul este la mutare;
- alți 6 indicatori care să precizeze dacă regii sau turnurile au mutat;
- ultima mutare a adversarului (pentru mutări „en-passant”).

Mutările se obțin prin adăugarea la poziția curentă a unei piese, a unui **increment** (pozitiv sau negativ), increment ce corespunde unei direcții de mutare (regele, dama, calul au 8 direcții, turnul, nebulul au 4 direcții etc.). La piesele cu bătaie lungă, incrementul pe o direcție se repetă pînă la întîlnirea unei piese sau a marginii tablei.

O poziție pe tablă este, de fapt, un indice în TABLA (120), iar incrementii se referă la incrementii de indice.

Cele mai mari dificultăți în programare le pune pionul, ținînd seama de neomogenitatea regulilor sale de mutare (transformare, „en-passant”, mutare pion 2 pași etc.).

Ceva dificultăți creează și tratarea rocadelor, dar aceste mutări speciale se pot asocia mutărilor regelui. Pentru a rezolva problema detecției dacă regele care mută prin rocadă nu trece prin cîmpurile atacate de adversar, unele programe, înainte de a genera lista mutărilor dintr-o poziție dată, calculează tabele paralele cu tabla de șah pentru a măsura „tăria” cu care cei doi jucători controlează cîmpurile tablei de șah. Aceste informații sînt utile pentru acordarea **ponderilor (notelor) statice** ale mutărilor.

Scrieți subprogramul de generare a listelor de mutări din poziții date, astfel încît să fie cît mai ușor de modificat, ținînd cont că acest subprogram va suferi cele mai dese schimbări în procesul evoluției programului dv.

**Arborele mutărilor.** Pentru a înțelege cum procedează un program de șah cînd calculează **mutarea optimă**, va trebui să facem o descriere a ceea ce înseamnă **arborele mutărilor**.

Într-o poziție dată pe tabla de șah, știînd cine este la mutare, se poate genera lista mutărilor posibile la acea mutare. Dacă luăm, pe rînd, fiecare mutare din listă și o efectuăm, obținem noi poziții pe tabla de șah, din care jucătorul advers poate genera cîte o nouă listă de mutări. Repetînd procesul de mai multe ori, alternativ pentru alb și negru, ne dăm seama că mulțimea de poziții care apar crește foarte mult, exponențial, funcție de numărul de **semimutări** pentru care se face generarea arborelui. Dacă admitem că într-o poziție de mijloc există circa 33 mutări, atunci chiar și pentru 3 mutări (6 semimutări) se generează un arbore cu un miliard de **noduri**, ceea ce este mult chiar și pentru calculatoarele rapide, căci un nod nu poate fi analizat doar cu o singură instrucțiune a calculatorului.

Să rămînem, deocamdată, în limitele ficțiunii, presupunînd că un calculator poate fi oricît de rapid. Cît de lungi pot fi ramurile în arborele de joc? Se știe că, prin repetări de mutări, se poate muta la nesfîrșit. Regulamentul jocului de șah asigură însă că arborele de joc să fie finit, căci sînt **remize** repetările de 3 ori a poziției, sau a lipsei unui schimb (sau a mutării unui pion) în mai mult de 50 de mutări.

Evident, **nodurile terminale** în arborele de joc sînt pozițiile de **mat** (regele advers **e în șah**) și orice ar muta va fi luat la următoarea mutare) sau **pat** (regele advers **nu e în șah**, și orice ar muta va fi luat la următoarea mutare).

Să remarcăm că, potrivit definiției de mai sus, pentru a vedea dacă o mutare este **mat** sau **pat**, mai este necesar să generăm încă două niveluri de semimutări, pînă cînd regele este luat efectiv. Prin urmare, pentru a rezolva un mat în 2 mutări, va trebui să generăm un **arbore (graf) de joc**, de 5 semimutări (3 + 2).

În figura 3 am reprezentat o poziție în care albul este la mutare și dă **mat** în 2 mutări.

8									
7									
6									
5									
4									
3									
2									
1									
	A	B	C	D	E	F	G	H	

fig. 3 Albul mută. Mat în două mutări.

(alb: rege H5, turn E5, cal F6;  
negru: rege H8, turn B4, pion G7)

În figura 4 am reprezentat un subarboare al arborelui de joc asociat poziției din figura 3.

Problema detecției faptului că o mutare este **mat** sau **pat**, creează dificultăți în cadrul unui program de șah, dificultăți pe care autorii de programe de șah le rezolvă în fel și chip:

— apelând subrutine complexe de verificare dacă poziția e mat sau pat;

— prelungind arborele cu încă două semimutări și urmărind dacă regele advers este luat, indiferent ce ar muta;

— tratări mixte, ca de exemplu în programul ATOM-64, în care se fac prelungiri de ramuri în arbore, doar când o mutare dă șah.

Cu excepția unor poziții cu puține piese, ori a unor probleme de mat în câteva mutări, este o utopie să se creadă că un calculator poate analiza toate ramurile până la nodurile terminale, asociate unei poziții de pe tabla de șah, chiar dacă s-ar aplica criteriul de reducere a variantelor (după cum se va vedea în expunerile viitoare).

Prin urmare, orice program de șah își va face o strategie în privința modului de a-și limita lungimea ramurilor pe care le generează, pentru a se încadra în anumite limite de timp.

**Ponderile statice.** Când generăm ramurile arborelui de joc, analizăm nodurile terminale, asociindu-le valori, numite **ponderi statice**, adică note pe care le putem calcula fără mutări. Toate remizele sînt la fel de bune, deci ponderile statice asociate lor sînt egale cu zero.

În privința pozițiilor de **mat** însă, orice jucător va muta astfel încît să dea **mat** în cît mai puține mutări și să primească **mat** cît mai tîrziu. Deci, dacă un mat se dă în  $x$  semimutări, atunci valoarea statică a acelei mutări este  $A-x$  (pentru alb) și  $-A+x$  (pentru negru).  $A$  este un număr foarte mare. Cu această precizare ne-am exprimat intenția de a folosi un sistem de **ponderi statice**, în așa fel, încît orice **avantaj** pentru alb să fie adăugat cu semnul „+” la ponderea statică, iar orice **avantaj** pentru negru se ia cu semnul „-”. Prin urmare, **penalizările** pentru alb și negru se consideră cu semnul „-”, respectiv „+”. Deci, albul tinde spre mutări cu ponderi cît mai mari, pozitive, iar negrul tinde spre mutări cu ponderi cît mai negative.

Mutările alternate ale albului și ale negrului conduc, în cele din urmă, la o luptă de tip **mini-max**, în care învinge cel care reușește să aducă pe adversar în zona convenabilă lui.

Într-un arbore limitat doar de regulamentul jocului de șah, nu am avea decît poziții de **mat** sau **remiză**, deci ponderile lor statice ar fi perfect determinate.

Ce se întîmplă însă într-un arbore cu ramurile trunchiate după un anumit număr de semimutări? Ce pondere statică trebuie acordată acestor poziții?

Răspunsul la această întrebare prezintă **problema cheie** a teoriei programării jocului de șah. Această problemă nu se poate rezolva îndeajuns de bine decît aplicînd felurite **criterii paleative** care încearcă să aproximeze ponderile statice. Pe de altă parte, se constată că, cu cît ramurile studiate într-un arbore de joc sînt mai lungi, cu atît efectul impreciziei evaluării ponderilor statice se atenuează.

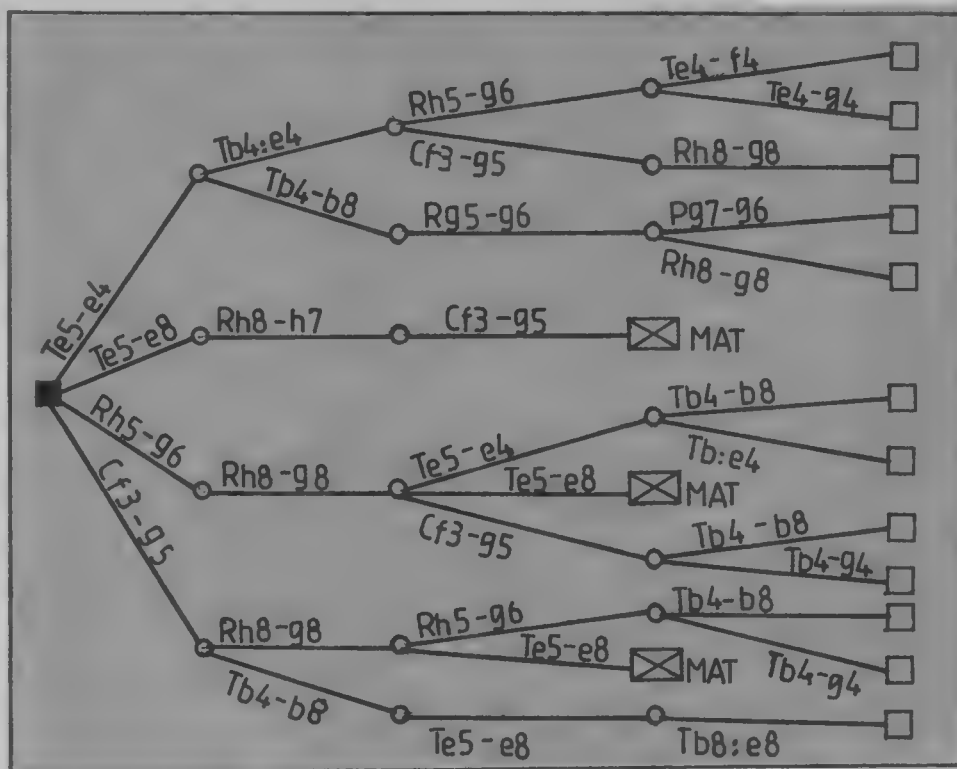


fig.4 O parte din arborele de joc asociat poziției din figura 3

De unde și tendința de a genera arbori cu variante cît mai lungi, fără a le răi însă într-atît încît să se piardă mutarea optimă.

Orice programator de șah se convinge, mai devreme sau mai tîrziu, că nu e bine ca arborele să aibă toate variantele de aceeași lungime. Justificarea acestei afirmații constă în aceea că, aprecierea mai bună a **ponderilor statice** se poate face în **poziții liniștite**, adică poziții în care nu sînt piese importante atacate (rege în șah, schimb de piese, fuga din șah, amenințări de furculițe, transformări etc.). De asemenea, dacă pe traseul unei variante au existat mutări forțate (schimburi, șah-uri) atunci variantele respective se cer prelungite cu numărul corespunzător de semimutări, căci altfel noile acțiuni gîndite (după respectivul schimb, șah succesiv) nu sînt matur judecate.

Evident, există tentația de ramificare cît mai bogată! Totuși programul trebuie să se încadreze în limita de timp alocată.

**Algoritmul mini-max.** Ce este mini-max? Este un procedeu matematic prin care se determină o **variantă optimă** în arborele de joc, ca rezultat al deciziilor de luptă ale ambilor jucători.

Prima mutare dintr-o **variantă optimă** este **mutarea optimă**. În orice nod în care **albul** este la mutare, el va alege mutarea sa corespunzătoare valorii **maxime** a **minimelor** pe care negrul le va putea alege ca mutare de răspuns. Când negrul este la mutare, el alege mutarea sa corespunzătoare valorii **minime** a **maximelor** pe care albul le va putea alege ca mutare de răspuns. Prin urmare, un proces **mini-max** este un proces recursiv, pe cîteva niveluri. Practic, procesul de evaluare a variantei **mini-max** este un proces care începe dinspre nodurile terminale ale arborelui de joc, spre baza arborelui.

Să considerăm că lungimea variantei celei mai mari în arbore este  $N$  și că în fiecare nod al arborelui se ajunge după  $l$  arce, adică  $l$  semimutări. Evident, mulțimea nodurilor arborelui se împarte în clase de echivalență, numite **niveluri**, dacă se consideră din clasa  $l$  nodurile în care se ajunge după  $l$  semimutări.

Procesul **mini-max** asociază fiecărui nod din arbore o **valoare mini-max** astfel:

— nodurilor terminale li se atribuie ca valori **mini-max** tocmai **ponderile statice**;

— se pleacă de la nivelul cel mai mare; în fiecare nod al unui nivel se consideră valorile **mini-max** ale mutărilor ce se generează din acel nod și se face cu ele maximum (minimum), dacă în acel nod la mutare este albul (respectiv negrul). Aceste maxime sau minime se asociază nodurilor corespunzătoare și, în momentul în care se îmbunătățește valoarea de maxim sau de minim, se reține și mutarea căreia îi corespunde, mutare ce se **asamblează cu subvariantele de continuare optimă** a acelei mutări, memorată anterior;

— se trece prin nivelul  $l-1$  și se repetă procesul **mini-max**.

Avertizăm cititorii că este mai ușor de păstrat **mutarea optimă** decît **varianta de joc optimă** („best line”). Nu este obligatoriu de menținut **varianta de joc optimă**, dar programele mai evaluate o fac și acest lucru le aduce utilități. Problema selecției acestor **variante de joc optimale** o vom mai relua.



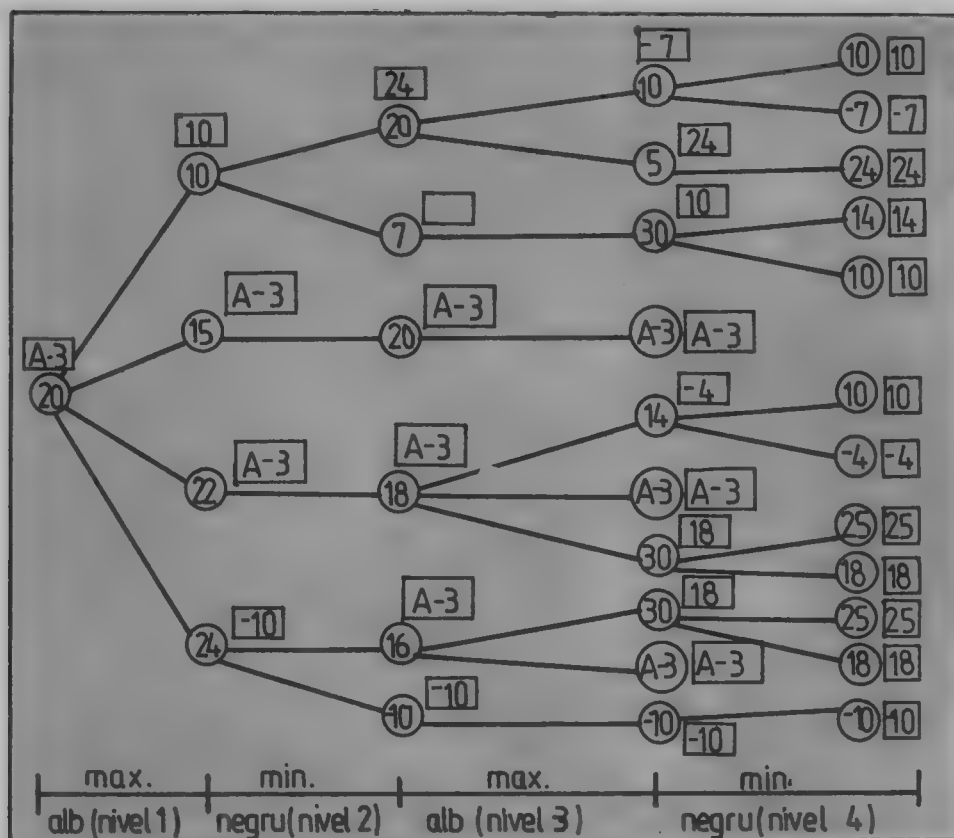


fig. 5 Ponderile statice (în cercuri) și valorile minimale (în pătrate) atribuite nodurilor arborelui de joc din fig. 4

În figura 5 este prezentat modul în care operează procesul mini-max.

### Mini-max, pro și contra.

De ce mini-max? E obligatoriu ca programarea șahului să fie formulată în termeni de variante în arborele de joc și mini-max?

Am auzit pe unii programatori care nu cunoșteau teoria mini-max programind șah. Evident, ei programau unele judecăți echivalente cu deciziile de alegere de mutări optime, conform mini-max.

Există obiecții multe privitoare la modelarea șahului în termeni de arbori de joc și mini-max. În mod natural, anumite grupuri de piese de pe tabla de șah interacționează mai puternic în cadrul grupurilor și mai slab între grupuri. Arborele de joc, unic pentru toate piesele, are neajunsul că proliferază în mod excesiv combinații nesemnificative între piesele diferitelor grupuri. Vă dați seama de neajunsul acestui fapt când lungimile ramurilor sunt limitate la câteva semimutări.

Această constatare l-a determinat pe M. Botvinnik, teoreticianul vestitului program PIONIER, să elaboreze așa numita teorie a zonelor de piese care interacționează între ele, precum și un mod de abordare a combinațiilor între zone (deci un sistem pe două niveluri). Din păcate, ideile lui Botvinnik, geniale, însă complicate pentru programare, nu s-au impus în lumea șahului programat deoarece sistemul suferea de unele rigidități și imperfecțiuni.

Însăși ideea de a privi variantele de joc sub forma unui arbore este criticabilă, căci, se știe, există și posibilitatea

ajungerii în aceleași poziții prin intervertiri de mutări. Adică modelul ar trebui să fie un graf orientat și nu un arbore. Totuși, intervertirile, procentual, nu sunt chiar atât de numeroase pe cât s-ar părea, în plus, gestiunea lor implică un mare consum de spațiu de memorare și timp de căutare, încât este mai convenabil să se renunțe la această gestiune.

### GENERAREA ARBORELUI DE JOC

Dacă în cele de mai sus am prezentat mai mult teoretic problema arborelui de joc și a procesului mini-max, acum să descriem câteva elemente concrete de programare.

Sunt multe modalități de abordare a generării de arbori de joc, de alegere a valorilor mini-max și de aplicare a criteriilor de reducere de variante (alfa-beta, alfa-beta-deep etc.) ce vor fi prezentate în următoarele expuneri. Nu vrem să impunem o conduită de programare (regretabil ar fi să-i îndrumăm pe cititorii noștri pe căi greșite), vrem ca această prezentare să constituie un material de referință.

Nu e bine ca un program de șah să genereze întâi de toate variantele de joc (de o anumită lungime) într-o zonă de memorie, ca apoi să înceapă să determine mutarea optimă, conform mini-max. Aceasta deoarece pe de o parte ar fi necesară o zonă foarte mare de memorie pentru reținerea tuturor variantelor, iar pe de altă parte s-ar genera toate variantele, inclusiv cele care n-ar mai fi necesare să fie generate conform unor criterii de reducere de variante.

Cele mai multe programe de șah adoptă o asemenea strategie de generare de variante de joc, încât ramurile să apară în memorie pe rând, într-o anumită ordine, iar operatorul mini-max să fie aplicat treptat pe fiecare din nivelurile de joc, pe măsură ce anumite ramuri ale arborelui sunt complet analizate. Într-o structură de liste de mutări (sub formă de stivă) este păstrată în memoria calculatorului varianta curentă din arbore ce se generează, precum și mutările adiacente variantei curente.

Mai explicit, pentru fiecare nivel 1, 2, 3, ... există câte o listă de mutări  $L(1)$ ,  $L(2)$ ,  $L(3)$ , ... a mutărilor adiacente variantei curente afecși o stivă de contexte asociată variantei curente, în care un element în stivă conține mai multe informații:

- indicator spre începutul listei  $L(i)$ ;
- pointer în interiorul listei  $L(i)$ , indicând mutarea din listă până la care s-a făcut analiza; (opțional)
- modificări pe tabla de șah pentru a da înapoi mutarea, pentru a ajunge la nivelul  $i-1$ ;
- valoarea mini-max parțială;
- alte informații.

În stiva contextelor fiecare pointer spre interiorul a câte unei liste  $L(i)$  indică tocmai mutarea care e făcută pentru a putea genera lista  $L(i+1)$ .

Listele  $L(i)$ , de la un anumit indice  $i$  încolo, conțin doar mutări ce vor face parte din variantele ce sînt preluate (schimburi, mutări forțate etc.). Nu are sens să fie trecute, în listele  $L(i)$ , mutările corespunzătoare nodurilor terminale ale arborelui, valorile statice ale acestor mutări terminale se consideră, direct, ca valori mini-max.

În momentul în care o subramură a grafului de joc este deja analizată, în acel moment se cunoaște valoarea mini-max, precum și varianta optimă de joc corespunzătoare acelei subramuri. Dacă această valoare mini-max îmbunătățește valoarea mini-max parțială pe nivelul imediat inferior, în acest moment se reține ca variantă optimă de joc varianta obținută prin concatenarea mutării curente cu varianta optimă de joc a subramurei. Se vede, deci, că pe fiecare nivel este păstrată câte o variantă optimă (parțială) de joc, care se îmbunătățește pe măsură ce ramurile devin analizate.

De asemenea, momentul îmbunătățirii valorii unui mini-max pe un nivel, este și momentul cînd se aplică diferite criterii de reducere a ramurilor în arborele de joc, cînd se observă că ramura analizată conține o soluție atât de bună, că nici nu mai are sens să fie studiate ramurile celorlalte mutări din listă, încă neanalizate.

**Algoritmul de calcul al mutării optime.** În fine, la sfîrșitul acestei prime părți a expunerii privind teoria programării șahului, prezentăm algoritmul de generare de variante, de alegere a mutării optime și a variantei optime de joc.

Fie  $M$  = lungimea propusă a variantei.

1. Inițializări:  $k = 1$ ;  $a = 0$  ( $a$  = nr. prelungiri).
2. Inițializarea valorii mini-max pe nivelul  $k$ .
3. Generarea listei de mutări  $L(k)$  în continuarea listei  $L(k-1)$ .
4. Dacă  $k$  este mai mare decît  $M$  sau  $k + a$  prea mare, sau regele advers e luat, înseamnă că încep să apară no-

duri terminale, deci se testează valoarea mini—max pe nivelul  $k$  în raport cu valorile pe nivelele inferioare, pentru a vedea dacă se poate aplica vreun criteriu de abandon al analizei mutărilor din lista  $L(k)$ .

Dacă criteriul se aplică, se face pasul 6.

Dacă nu, se continuă cu pasul 5.

5. Se alege mutarea cu ponderea statică cea mai bună din lista  $L(k)$ . Dacă există, mutarea se suprimă din listă, se efectuează pe tabla de șah, ținând la zi informațiile de context pe nivelul  $k$  (PLAY).

Salvează și valoarea  $a$  în contextul stivei.

Dacă mutarea e un schimb, un șah, sau răspuns la șah etc., atunci  $a=a+1$ . Se face  $k=k+1$  și salt la pasul 2.

Dacă  $L(k)$  devine o listă vidă, se face pasul 6.

6. Valoarea mini—max (complet definită) pe nivelul  $k$  se compară cu valoarea mini—max (parțială) pe nivelul  $k-1$ . Dacă nu are loc o îmbunătățire a valorii mini—max, se trece la pasul 9.

7. În cazul îmbunătățirii valorii mini—max, pe nivelul  $k-1$  se reține o variantă optimă obținută prin concate-

narea mutării de trecere de la nivelul  $k-1$  la nivelul  $k$ , cu varianta optimă de la nivelul  $k$ .

8. De asemenea, cînd se îmbunătățește valoarea mini—max pe nivelul  $k-1$  atunci se aplică un criteriu de reducere a variantelor (abandon al analizei mutărilor din lista  $L(k-1)$  care au mai rămas) și se trece la pasul 10. Dacă criteriul nu funcționează, se continuă cu pasul 9.

9.  $k=k-1$ . Dacă  $k$  e mai mic decît 0, arborele a fost analizat în întregime, se cunoaște valoarea mini—max și varianta optimă de joc (best line).

Dacă  $k$  e mai mare ca 0, atunci se revine la contextul valorilor stivei de la nivelul noului  $k$  și se trece la pasul 5 (REPLAY).

10.  $k=k-1$ . Se reface contextul stivei la nivelul noului  $k$ . Se trece la pasul 9.

## COMENTARII

— În limbajele C, PASCAL, care admit apeluri recursive de proceduri, lista  $L(k)$ , precum și informațiile privind un nivel, se pot defini ca variabile

locale ale procedurilor, ce se alocă în regim de stivă;

— Testele de la pasul 4 se fac, de fapt, în rutina care generează lista de mutări  $L(k)$ , pe măsură ce o mutare este generată;

— Se observă că algoritmul, la pasul 5, nu face întîl o sortare completă a listei  $L(k)$ , ci doar ia cîte o mutare, cea mai bună din cele rămase în  $L(k)$ ; sînt dese cazuri cînd lista  $L(k)$  nu va fi studiată completă (apar reduceri de ramuri), deci o sortare completă a lui  $L(k)$  nu este necesară;

— Alegerea mutărilor cu ponderea statică cea mai bună din lista  $L(k)$ , face ca ramurile arborelui să fie generate în ordinea cît mai plauzibilă de a fi optime, mutările bune găsire de la început au un pronunțat efect de tăiere a ramurilor adiacente care au mai rămas; astfel, se vede că aprecierea cît mai corectă a ponderilor statice are un rol însemnat atît în acuratețea valorii mini—max stabilite, cît și a vitezei de analiză a arborelui.

Vom reveni cu alte considerații teoretice asupra programării șahului, în special asupra criteriilor de reducere de variante și de evaluare de ponderi statice.

## În dezbatere:

## INFORMATICA

## ÎN ROMÂNIA

## ANCHETA

**I**ată fiind importanța acestui domeniu atît în viața socială, cît și în cea economică, redacția noastră a inițiat o anchetă care își propune doar să ridice niște probleme, în concordanță cu opiniile celor care au avut amabilitatea să ne răspundă, și nu să dea soluții definitive. De altfel, ancheta noastră rămîne deschisă, vom mai reveni asupra ei, deoarece subiectul nu poate fi epuizat într-un număr de revistă și nici nu ne propunem de fapt acest lucru. Scopul nostru final este sensibilizarea unor factori de decizie, deschiderea unor dezbateri constructive, trezirea intereselor — în ultimă instanță — pentru o disciplină — *informatica* — și un instrument de lucru — *calculatorul* — fără de care nu poate exista nici o economie modernă. Am mai inițiat această anchetă și pentru faptul că țara noastră a avut matematicieni și cercetători de frunte care și-au adus o contribuție deosebită la fundamentele teoretice ale informaticii și de ale căror nume se leagă realizări notabile de-a lungul vremii, realizări ce ne-au situat acum vreo trei

decenii între țările cu mari perspective în domeniu. Ceea ce a urmat se cunoaște și nu are rost să mai revenim. Important este acum modul în care se abordează acest moment al informaticii românești, pentru că această clipă trebuie fructificată la maximum. Și pe cei reticenți la astfel de anchete și, mai ales, la eficacitatea lor îi rugăm să vadă în această anchetă atît sensul ei optimist, de încredere că specialiștii români vor depăși acest moment dificil și vor recupera anii ce au trecut, precum și dorința noastră de a sprijini, cu propriile mijloace, dezvoltarea informaticii în România.

Desigur că *informatica* este în prezent caracterizată printr-o accentuată interdisciplinaritate, deci este puternic condiționată de aplicații. Ca atare, dezvoltarea informației în țara noastră este intim legată și condiționată de dezvoltarea economiei în ansamblu. Noi am extras din aceasta *informatica*, pentru a încerca determinarea unor posibile soluții pe multiple planuri, soluții menite, în baza acelorași legături amintite mai sus, să contribuie la dezvoltarea economiei în ansam-

blul ei.

Așadar, am adresat diferitelor personalități, cadre didactice universitare, specialiști din institute de cercetări și din producția de profil; cadre didactice din învățămîntul preuniversitar etc., următoarele trei întrebări:

1) În ce direcții ar trebui dezvoltată producția românească de calculatoare și echipamente periferice?

2) Care ar fi direcțiile prioritare de dezvoltare a soft-ului și dacă acestea ar trebui sau nu să fie în concordanță cu hard-ul ce se produce în țară?

3) Care ar fi dotările minimale în școli și de la ce vîrstă pentru cele două direcții distincte ale învățămîntului asistat de calculator: *informatica în învățămînt* și *informaticizarea învățămîntului*?

● În ceea ce privește răspunsurile la prima întrebare, se impun deja cîteva concluzii, unele sintetizate foarte bine în două dintre răspunsurile primite:

„Vom face adaptări, asimilări ale produselor performante, care sînt cu două generații înainte față de ceea ce se produce la noi, și, ceea ce este poate mai important, să ajungem să posedăm tehnologia la zi pentru aceste produse. Nu cred că trebuie să ne preocupe prea mult producerea calculatoarelor mari — main-frame —, ci producerea de mini și mai ales micro-uri. Pentru utilizări de înaltă profesionalitate trebuie abordat și domeniul micro-urilor dedicate aplicațiilor sub forma stațiilor de lucru. Nu trebuie uitat microcalculatorul ca bun de larg consum, sub nivelul de PC ca preț și posibilități, utilizat în familie pentru divertisment și instruire ca un calculator de casă (home computer) sau ca un joc pe televizor (TV game) care la noi este foarte puțin răspândit. În privința echipamentelor periferice electronice (terminale, blocuri de măsură electronice etc.), acestea pot fi aduse prin efort intern la performanțe competitive. Pentru perifericele de natură electromecanică (imprimante, plottere, discuri magnetice de diferite tipuri) ar trebui reținute doar acelea la care se poate elabora sau cumpăra tehnologia la zi a părților mecanice”.

„Cheia succesului constă în specializarea strictă de nivel cît se poate de înalt. Cred că pentru industria de calculatoare direcțiile principale de dezvoltare trebuie să fie reprezentate de sisteme de calcul medii-mari, de mini și microcalculatoare pe 16 și 32 de biți, cît și de calculatoare personale pe 8 biți.

Evident, nu trebuie neglijate nici necesitățile stringente ale informaticii românești la ora actuală privind echipamentele periferice: unități de memorie cu discuri magnetice de tip rigid și flexibil, plottere, imprimante cu laser, stații grafice cu rezoluție ridicată și videoterminal etc.”

În general, putem să afirmăm că răspunsurile au fost orientate spre dezvoltarea microcalculatoarelor și a „personalelor” ca bunuri de larg consum; spre programe ușor de utilizat de către neinformaticieni, alinierea la standardele internaționale pentru diferitele familii de calculatoare (IBM, DEC, pentru micro și respectiv minicalculatoare), cooperarea cu firme străine, dezvoltarea rețelelor de calculatoare și a stațiilor de lucru CAD/CAM. În ceea ce privește echipamentele periferice, se desprind cîteva tendințe clare: memorii magnetice (cu capacitate mare — atît pe linia discurilor, cît și a benzilor magnetice streaming), imprimante rapide, display-uri color de înaltă rezoluție.

● La cea de-a doua întrebare se desprind de asemenea cîteva con-

cluzii interesante. Axa centrală a majorității răspunsurilor se referă la soft-ul de aplicații care poate să se constituie într-o premisă de bază a unei industrii naționale de soft. Un alt aspect al dezvoltării soft-ului este acela că nu trebuie legat intrinsec de dezvoltarea industriei autohtone, ci poate constitui baza unei activități care să ofere servicii sub forma aplicațiilor diverse. Așadar, „Ingineria software poate deveni o componentă extinsă și cu personal foarte numeros în cadrul unei industrii naționale de tehnică de calcul”. Aceasta presupune utilizarea programelor standard pentru CAD/CAM etc., care constituie de asemenea un posibil mod de aliniere la cerințele actuale ale pieței mondiale, în diferite domenii. Ar trebui în mod esențial creat cadrul legal care să pună soft-ul în drepturile sale, deoarece el constituie o importantă resursă economică.

O idee interesantă este aceea a „caselor de soft” care se pot dezvolta fie în cadrul institutelor de cercetare și universităților (acest lucru ni se pare esențial, și anume implicarea reală a studenților și cadrelor didactice în activitatea de cercetare), fie particulare. Ca o posibilă concluzie, cităm un fragment dintr-un răspuns:

„Produsele software de aplicație vor constitui un cîmp de activitate cu deschidere largă și care pot forma un segment foarte important al activității pentru o industrie națională de software. Producția de software de anvergură industrială trebuie pornită cît mai repede, deoarece necesită o investiție materială relativ redusă în raport cu valoarea produselor realizate și este tot atît de importantă ca și producția de echipamente”.

● În legătură cu cea de-a treia întrebare disputele sînt foarte numeroase. Această problemă a mai făcut obiectul a numeroase anchete, deoarece „școala la ora informaticii” se anunță a fi o problemă foarte complexă, cu multiple fațete și cu implicații deosebite; tocmai de aceea, ne-am gîndit ca pentru început să acordăm un spațiu mai mare răspunsului pe care ni l-a dat un cadru didactic din învățămîntul liceal, din care se pot trage foarte ușor concluziile:

„În învățămîntul preuniversitar, informatica trebuie să pătrundă diferențiat:

— în școli profesionale și licee elevii să-și însușească un bagaj minim de cunoștințe care să le permită o adaptare rapidă la tehnica de calcul cu care vor lucra după absolvire;

— să se mențină liceele de informatică, în scopul formării specialiștilor cu studii medii. Aceste licee să funcționeze ca liceu teoretic real cu specialitate informatică.

Necesitatea liceelor de informatică decurge din următoarele motive:

— informatica se învață bine începînd cu vîrsta de 15 ani, cînd puterea de imaginație este maximă;

— schimbările care intervin în domeniul tehnicii de calcul sînt atît de rapide încît nu le poți stăpîni cu adevărat printr-o pregătire particulară de cîteva luni. Însăși programa școlară pentru obiectul informatică trebuie schimbată și actualizată aproape anual;

— specializarea în informatică determină implicit crearea unei culturi generale vaste. Atenția cercetărilor din domeniul informaticii și profesorilor de informatică, a pedagogilor și psihologilor trebuie concentrată în următoarele direcții: construirea unor „baze de cunoștințe” care să înmagazineze un număr mare și variat de cunoștințe referitoare la diverse domenii, ușor prelucrabile prin dezvoltarea unui software bogat și puternic; utilizarea calculatoarelor în procesul de evaluare a cunoștințelor prin construirea unor teste, pentru diverse discipline de învățămînt”.

Deci calculatorul trebuie să pătrundă în școală alături de profesor. Aceasta implică însă mai multe aspecte: o bază materială (deci o industrie care să asigure dotarea tuturor școlilor cu calculatoare), adecvată desfășurării în condiții normale a procesului didactic, dotare ce ar trebui să fie diferențiată în funcție de nivelul școlar (microcalculatoare pe 8 biți din gama HC, COBRA etc. în ciclul primar, gimnazii sau unele licee și calculatoare personale pe 16 biți din gama PC XT/AT în licee de profil și facultăți), cadre care să aibă pregătirea necesară predării informaticii, programe care să fie în primul rînd pedagogice și eficiente pentru a întregi actul predării lecțiilor și învățării, disocierea clară a celor două tendințe: informatica în învățămînt — prin menținerea și mărirea numărului de licee de informatică din țară — și informatizarea școlii, folosirea potențialului uman de inteligență al copiilor avînd în vedere rezultatele pe care ei le obțin la competițiile internaționale. Într-o primă etapă, deoarece multe dintre punctele sesizate în răspunsuri presupun rezolvări pe termen destul de lung, cercurile de informatică din școli (la toate nivelurile), patronate de institute de profil și de întreprinderi producătoare, vor pregăti momentul trecerii la informatizarea învățămîntului.

Ancheta noastră nu s-a încheiat. Sintetizînd unele aspecte și neglijînd poate altele, ea nu a dorit decît să deschidă o discuție la care așteptăm în continuare opiniile dumneavoastră.

Anchetă realizată de MIHAELA GORODCOV



Iată o reușită, din toate punctele de vedere, a specialiștilor în domeniu din țara noastră! Menționăm faptul că materialul ne-a fost remis la redacție în limba engleză. Noi, din motive multiple, vă vom prezenta doar un rezumat în limba română, iar materialul original a fost trimis prin rețeaua IDG, sub antetul „Contribuții”, la International Data Communication Group. Pe această cale, invităm pe toți colaboratorii, cercetătorii și utilizatorii de sisteme de calcul să ne remită astfel de materiale spre a fi difuzate în rețeaua IDG.

## 1. Precizări introductive

**ELECT'90** este numele proiectului care a avut drept scop procesarea datelor pe calculator referitoare la alegerile din 20 mai 1990 din România. Proiectul a început în martie, după aprobarea legii electorale, și s-a încheiat pe 25 mai, o dată cu anunțarea oficială a rezultatelor alegerilor.

Derularea proiectului a avut în vedere următoarele: în fiecare capitală de județ a fost stabilit câte un birou local pentru alegeri (District Electoral Office-DEO), în total 41, precum și un birou central electoral în București (Central Electoral Office-CEO). Din partea guvernului a fost însărcinată Comisia Națională de Statistică - CNS, pentru a duce la bun sfârșit proiectul. Aceasta, la rândul ei, a ales ITC-București ca proiectant general și executant pentru partea de software a acestui proiect și Rom Control Data Ltd. ca producătoare hardware.

Citeva date statistice: ● 12394 centre de votare în toată țara; ● în cursa electorală au intrat 71 de partide politice; ● orașul București a fost o excepție de la Legea electorală cu 1 030 centre de votare, 102 liste de candidați pentru funcțiile de deputați și 66 de liste pentru senatori.

## 2. Hardware

Pentru a duce la bun sfârșit proiectul **ELECT'90** a fost implementată, în timp record, o rețea de nivel național. Rețeaua a constat în 41 de noduri în fiecare DEO și o rețea de tip LAN (Local Area Network) la CEO. Nodurile erau formate din 2-4 PC Compatibile AT (12 MHz, 80286 CPU, 1MB RAM, 3 1/2 floppy disk drive, 40/80 MB hard disk drive și monitor color EGA), 2 imprimante matriciale, precum și cel puțin 1 modem inclus.

Rețeaua LAN de la CEO a fost concepută cu două servere de tipul 386 SX/33 MHz cu 330 MB capacitate în hard disc. A fost utilizat pachetul de programe NOVELL 2.15 pentru rețele precum și adaptoarele LAN ARCNET cu viteză de 2 500 biți/secundă. Un număr de 7 PC/AT au asigurat comunicația la 2 400 bauds, fiecare AT fiind conectat cu 6 județe.

Alte două microcalculatoare, de asemenea, compatibile PC/AT, au asigurat prezentarea grafică (parțială și

finală) a rezultatelor, unul dintre acestea fiind în permanență conectat la un car TV pentru transmisii. În total, acest proiect a inclus 150 echipamente (calculatoare, imprimante, modemi etc.). În cadrul CEO s-au utilizat, de exemplu, două imprimante laser de tipul Hewlett Packard, sisteme conectate la server.

## 3. Software

Proiectul **ELECT'90** a avut următoarele scopuri:

- fiabilitate maximă a hard-ului;
- precizie absolută a datelor prelucrate și a rezultatelor finale sau parțiale;
- protecția datelor în cazul defecării hard-ului prin salvarea periodică a acestora pe suporturi externe (floppy discuri sau hard discuri);
- interfață prietenoasă cu utilizatorul;
- prezentarea grafică a rezultatelor parțiale și finale pe monitoare TV.

Pentru a mări viteza de introducere a datelor s-au utilizat două terminale compatibile DEC VT 100 cu interfață serială.

Funcțiile pachetelor de programare pentru DEO sînt următoarele:

- 1.— introducerea datelor (cu toate aspectele legate de fiecare birou de votare, număr de electori, voturi acumulate, număr de voturi pentru fiecare listă de candidați etc.);
- 2.— includerea în baza de date a înregistrărilor rapoartelor oficiale;
- 3.— autentificare;
- 4.— corecția bazei de date;
- 5.— comunicarea datelor-care au presupus transmiterea rezultatelor parțiale și finale către CEO;
- 6.— actualizarea conținutului discurilor între mașinile din cadrul aceleiași DEO;
- 7.— tipărirea datelor și controlul dispecerului de tipărire;
- 8.— calculul electoral propriu zis;
- 9.— verificarea bazei de date.

După colectarea datelor de la fiecare DEO, în cadrul CEO echipamentele specializate în comunicații de date, recepționează rezultatele parțiale și le transferă la serverul de fișiere prin intermediul LAN, în directorul specific alocat pentru fiecare județ, precum și în directorul special de evidență. Periodic, baza de date este împăștată cu cele mai recente rezultate parțiale din centrele județene.

Toate programele de aplicație au

fost concepute în C, PASCAL și ASSEMBLER.

Pentru pregătirea personalului implicat în acest proiect s-a organizat un seminar cu durată de 3 zile în București.

În ceea ce privește atât hard-ul, cât și pachetele software, pe parcursul derulării proiectului nu au fost înregistrate incidente majore. Echipa formată din tineri specialiști de la ITC, cu o medie de vîrstă de 30 de ani, a demonstrat — pe parcursul a celor 50 de zile, cât a durat proiectul **ELECT'90** — că este capabilă de performanțe de nivel european.

În încheiere se aduc mulțumiri tuturor factorilor de decizie și răspundere, în special Comisiei Naționale de Statistică, echipei de instalare și asistență tehnică de la Rom Control Data Ltd.; precum și colegilor de la Centrele Teritoriale de Calcul Electronic și din ITC pentru efortul depus la ducerea la bun sfârșit a acestui extrem de interesant proiect, derulat în premieră în țara noastră și încheiat cu mult succes în mai 1990.

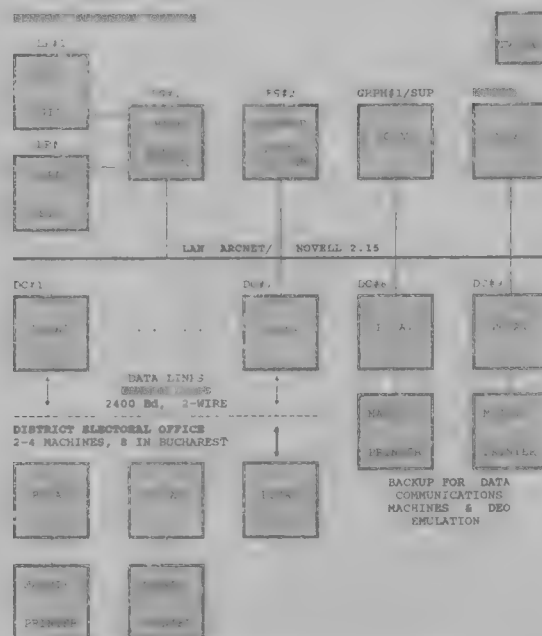


FIG. 1. HARDWARE CONFIGURATIONS OF ELECT'90 PROJECT

# ELEMENTE PRACTICE DE GRAFICĂ EGA/VGA

Într-un articol din numărul trecut au fost trecute în revistă o serie de principii ce stau la baza cuploarelor grafice utilizate în calculatoarele compatibile IBM PC/XT/AT sau PS/2. Vom încerca acum să aprofundăm aceste noțiuni prin examinarea în detaliu a posibilităților practice de implementare a unor funcții grafice de bază la nivel de coordonate fizice ale cuplorului **Device Coordinates**. Accentul va fi pus pe cuplorul de tip EGA lucrând în modul 640x350 cu 16 culori simultane. Extinderea la VGA în modul 640x480 cu 16 culori simultane este banală, iar în modul 320x200 cu 256 de culori simultane tratarea problemelor este, în mod surprinzător, mult mai simplă.

Scopul final este ca dumneavoastră să puteți implementa o bibliotecă de funcții grafice. Pentru ușurința lucrului am ales un limbaj de nivel înalt de mare popularitate și anume C, astfel încât efortul de programare să fie minim, iar întreaga dumneavoastră atenție să poată fi focalizată asupra problemelor de grafică. Odată depășită etapa de implementare se poate trece la optimizare prin rescrierea unor proceduri în limbaj de asamblare, obținându-se astfel sporuri de viteză între 5 și 50%.

Probabil că prima întrebare pe care o puneți este: de ce să construim o

nouă bibliotecă în condițiile în care compilatoarele actuale posedă deja una? Motivele sînt două:

— ele nu răspund decât în mică măsură necesităților de realizare a unor efecte grafice nestandard;

— nu asigură instrumente pentru realizarea interactivității cu utilizatorul.

Deoarece cuplorul EGA standard are porturi WRITE-ONLY starea lui este memorată într-o copie RAM a bibliotecii, copie a cărei poziție și structură pot fi cu greu găsite. Orice artificiu realizat direct asupra cuplorului va duce la o incoerență între starea reală și copia RAM, consecințele putînd fi imprevizibile. Deci pentru orice aplicație ce necesită facilități inexistente în bibliotecile compilatoarelor este necesar să ne bazăm pe un produs aflat sub controlul nostru total.

În cazul limbajului C, din considerente de dezvoltare și exploatare vom utiliza un grup de mai multe fișiere cu următoarele semnificații: LIB.H — header cu definiri de constante și tipuri; LIB-BAS.C — funcții de control al regimului de lucru; LIB-DC.C — primitive grafice la nivel Device Coordinates; LIB-BLK.C — funcții de lucru cu blocuri de imagine; APL.C — fișier „aplicație” în care vom exersa utilizarea funcțiilor. Acest fișier nu va fi in-

tr dus în cadrul bibliotecii (GRF.LIB).

„Aplicația” pe care noi am construit-o pentru dezvoltarea și verificarea funcțiilor grafice a fost un editor grafic de mică complexitate. Pe acesta nu-l vom prezenta pentru a nu abate atenția de la problemele de implementare a funcțiilor grafice pentru cuplorul EGA/VGA. El nu face altceva decît să apeleze funcțiile grafice la comenzi ale unor taste (în regim „hotkeys”) cu parametri luați din context sau de la claviatura numerică. Acest lucru îl puteți realiza și dumneavoastră foarte simplu și într-o infinitate de variante.

Pentru a completa contextul de lucru, în cazul în care utilizați compilatorul Turbo C, mai este necesar să realizați și un „project file”, denumit GRF.PRJ cu următorul conținut:

```
LIB-BAS.C (LIB.H)
LIB-DC.C (LIB.H)
LIB-BLK.C (LIB.H)
APL.C (LIB.H)
```

Introduceți numele acestui fișier la opțiunea PROJECT și salvați configurația curentă a compilatorului.

Aceste detalii preliminare fiind lămurite, să trecem efectiv la problemele de grafică.

Primele informații pe care trebuie să le introducem în header sînt cele referitoare la adresa de început a memoriei ecran și adresele porturilor EGA.

## LIB.H:

```
#define EGA_MEM 0xa0000000
#define P_seq_s 0x3c4 /* Sequencer select */
#define P_seq_d 0x3c5 /* Sequencer data */
#define P_ctr_r 0x3cf /* Controller register */
#define P_ctr_a 0x3ce /* Controller address */
#define P_atpar 0x3da /* Attribute/Palette reset */
#define P_at_pa 0x3c0 /* Attribute/Palette */
#define P_crt_a 0x3d4 /* CRT Controller Address */
#define P_crt_r 0x3d5 /* CRT Controller register */
#define RES_X 80 /* Nr. bytes/line */
```

Asupra ultimei constante dorim să atragem atenția că oricare ar fi modul de lucru (alfanumeric sau grafic) totdeauna pe o linie sînt baleiați 80 de octeți (excepție: modul 13H la VGA).

Prima funcție pe care trebuie să o

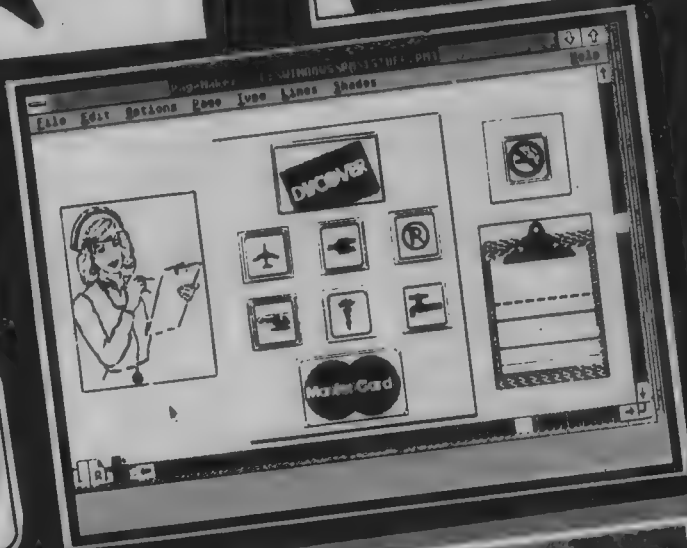
implementăm este aceea de trecere în mod grafic. Practic, înseamnă a programa porturile cuplorului cu valorile cuprinse în PROM-ul adițional. Cea mai simplă metodă este să utilizăm direct o funcție BIOS. În același timp e

bine să schimbăm și setul de caractere la versiunea 8 x 8 (43 de linii). De asemenea, tot în cadrul acestei funcții vom defini și inițializa un pointer deosebit de important a cărei utilitate se va vedea ulterior.

ACTUALITATEA PC

JERSEA

DISCOVER



MasterCard™





```
LIB_BAS.C:
#include <dos.h>
#include "lib.h"
union REGS *preg, reg;
char far *p_ega;

gopen()

{
    preg = &reg;
    preg->x.ax = 0x0010;
    int86(0x10, preg, preg); /* 640*350 */
    preg->x.ax = 0x1123;
    preg->h.bl = 0x03;
    preg->h.dl = 43;
    int86(0x10, preg, preg); /* 43 linii 8*8 */
    p_ega = MK_FP(0xa000, 0);
}
```

Tot în acest fișier se găsește și funcția complementară care asigură revenirea în mod alfanumeric:

```
LIB_BAS.C:
gclose()
{
    preg->x.ax = 0x0002;
    int86(0x10, preg, preg);
}
```

O problemă ce a fost omisă în funcția gopen( ) este aceea a depistării cu-

plurului EGA/VGA. Soluții sînt multiple, dar o versiune simplă este să verificăm dacă la adresa 0000:0487 avem marca 0x60.

## CULORI

Odată intrați în mod grafic ne vom gândi imediat la implementarea unor primitive grafice: line, polyline, circle, rectangle etc. Înainte de aceasta trebuie să ne aducem aminte că ele sînt caracterizate printr-o serie de atribute, dintre care cel mai important este culoarea. Dacă vom reciti articolul din numărul trecut vom vedea că stabilirea unei culori de trasare se poate face în două moduri:

— stabilirea unui index în LUT (Look-Up-Table);

— stabilirea proporțiilor RGB pentru indexul curent în LUT.

Deoarece primitivele sînt diferite, iar unele au mai mult de un atribut de culoare, așa cum vom vedea mai târziu, nu putem să realizăm atacul direct al porturilor cuplorului. Vom atașa fiecărei primitive grafice o imagine memorie a atributelor de culoare pe care o vom folosi pentru programarea cuplorului numai în momentul trasării. Deci setarea unui atribut de culoare se reduce la înscriserea unei variabile interne a bibliotecii, transparenta pentru utilizator.

Al doilea mod de stabilire a culorii de trasare atacă direct cuplorul, modificînd procentele RGB la indexul de intrare în LUT preluat din variabilele interne. Astfel pixelii de pe ecran trasați cu un anumit index își vor modifica culoarea fizică (vezi în numărul trecut legătura între vuloarea logică - index - și fizică - RGB - stabilită prin intermediul LUT). Pentru a putea păstra o compatibilitate între EGA și VGA valorile RGB nu vor fi date în binar ci vor fi normalizate 00...10. Funcțiile pentru VGA pot fi ușor adaptate la o paletă de 256 k culori.

Se observă că din același motiv al imposibilității citirii porturilor EGA (modele mai vechi) am introdus și o copie în memorie a LUT (Look-Up-Table).

Pentru cei ce doresc în mod efectiv să implementeze pas cu pas rutinele din acest articol atragem atenția că **declarațiile variabilelor globale trebuie plasate înaintea declarațiilor de funcții și declarate externe în celelalte module.**

Următoarele funcții realizează efectiv comanda porturilor EGA/VGA și dacă este să ne păstrăm în standardul CGI (Computer Graphics Interface) ele trebuie să fie interne bibliotecii (transparente utilizatorului).

```
LIB_BAS.C:
unsigned char color, bk_color, per_color;
unsigned char LUT[16];
background_color_index(index)
    unsigned char index;
    {
        bk_color = index;
    }

background_color(red, green, blue)
    float red, green, blue;
    {
        set_LUT_color(bk_color, RGB_to_mask(red, green, blue));
        LUT[bk_color] = RGB_to_mask(red, green, blue);
    }

line_color_index(index)
    unsigned char index;
    {
        color = index;
    }
```

```

line_color(red,green,blue)
    float      red,green,blue;
    {
        set_LUT_color(color,RGB_to_mask(color,red,green,blue));
        LUT[color] = RGB_to_mask(color,red,green,blue);
    }

```

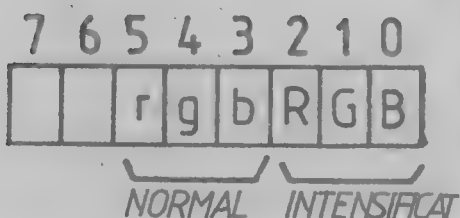
Acesta este punctul în care VGA poate oferi incomparabil mai multe posibilități, și anume o paletă de 256 k culori! Aceasta se realizează în modul 12H, mod în care următoarele rutine devin specifice VGA și mult mai complicate. Din aceste considerente de parcă de calculatoare existent în România, ne vom rezuma totuși la un set de rutine compatibile mod 10H EGA (640x350) cit și mod 12H VGA (640x480) în care vom afișa 16 culori simultan dintr-o paletă de 64.

```

LIB_BAS.C:
set_LUT_entry(index)
unsigned char index;
{
    outp(P_ctr_a,0x01);
    outp(P_ctr_r,0xff);
    outp(P_ctr_a,0);
    outp(P_ctr_r,index);
    outp(P_ctr_a,0x01);
    outp(P_ctr_r,0x00);
    outp(P_ctr_a,0x00);
    outp(P_ctr_r,0x00);
}
set_LUT_color
(index,mask)
unsigned char
index,mask;
{
    inp(P_atpar);
    outp(P_at_pa,
        index & 0x0f);
    outp(P_at_pa,mask);
    inp(P_atpar);
    outp(P_at_pa,0x20);
}

```

Următoarea rutină de conversie de la valori normalizate RGB la mască de biți se bazează pe următoarea structură a unei intrări în LUT:



```

LIB_BAS.C:
RGB_to_mask(red,green,blue)
    float      red,green,blue;
    { unsigned char mask;
      mask = 0x00;
      if(red <= 0.25)
          ;
      else if(red <= 0.5)
          mask |= 0x20;
      else if(red <= 0.75)
          mask |= 0x04;
      else
          mask |= 0x24;
      if(green <= 0.25)
          ;
      else if(green <= 0.5)
          mask |= 0x10;
      else if(green <= 0.75)
          mask |= 0x02;
      else
          mask |= 0x12;
      if(blue <= 0.25)
          ;
      else if(blue <= 0.5)
          mask |= 0x08;
      else if(blue <= 0.75)
          mask |= 0x01;
      else
          mask |= 0x09;
      return mask;
    }

```

Culoarea nu este însă un atribut doar al background-ului și foreground-ului. Pentru cuploarele EGA/VGA în afara ariei de pixeli există un chenar a cărui culoare nu

este legată de nici o intrare în LUT și poate fi specificată direct. Privind în corespondență cu semnalul TV ea corespunde zonei de blanking și în cazul nostru poartă denumirea de overscan.

```

LIB_BAS.C:
overscan_color(red,green,blue)
float      red,green,blue;
{
    inp(P_atpar);
    outp(P_at_pa,0x11);
    outp(P_at_pa,RGB_to_mask(red,green,blue));
    inp(P_atpar);
    outp(P_at_pa,0x20);
}

```

Vom trece acum la delicata problemă a trasării unei primitive grafice. Pentru a înțelege mai întâi mecanismele de utilizare a registrelor EGA/VGA vom studia un caz unde nu intervine nici un algoritm de trasare și anume „aprinderea” unui pixel pe ecran. Aceasta implică două probleme:

- setarea culorii;
- stabilirea corespondenței între poziția geometrică exprimată prin coordonatele (x, y) și adresa fizică a memoriei de afișare.

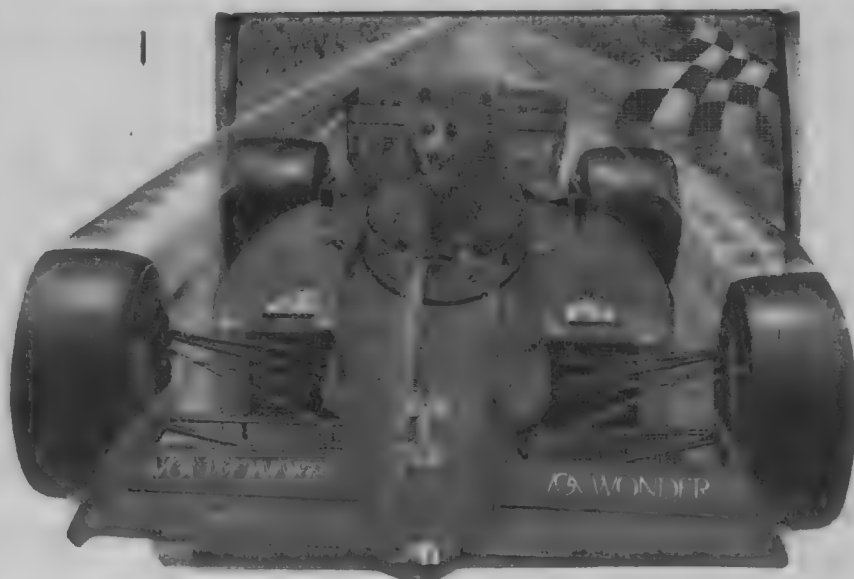
Așa cum am arătat în numărul trecut, memoria video are o structură spațială. Procesorul calculatorului nu are acces decât pe suprafața x, y și chiar și acolo numai trecând prin ALU și o mască. Pentru lucrul în adâncime, deci cu indexul de culoare, nu se lucrează decât prin intermediul unei perechi de registre care se găsesc la index 0 și 1 în Graphics Controller. În registrul 0 se scrie practic culoarea logică de trasare, adică codul uneia din cele 16 intrări în LUT. În momentul activării unui pixel, în fiecare plan de culoare se înscrie bitul corespunzător din acest registru. În realitate situația este puțin mai complicată fiindcă apare o validare suplimentară prin registrul 1. Astfel, un 1 în registrul de validare va permite înscrierea planului cu valoarea corespunzătoare, iar un 0 în registrul de validare va lăsa conținutul neschimbat din punctul de vedere al culorii.

Nici cea de-a doua problemă nu este simplă. Din punct de vedere geometric un pixel se găsește într-un spațiu cu două dimensiuni. În marea majoritate a modurilor grafice ale EGA/VGA unui pixel îi corespunde în fiecare plan de culoare un bit. Din nefericire bitul se găsește într-un spațiu cu alte proprietăți organizat unidimensional (linear de la adresa a000:0000) și discretizat în cuante numite octeți (bytes).

Deci înainte de orice operație de trasare trebuie să efectuăm o transformare din spațiul geometric de coordonate (x, y) în spațiul de memorie byte (relativ la începutul memoriei ecran) și poziție în interiorul byte-ului și să lucrăm folosind acest nou sistem de coordonate. Din păcate nu este suficient fiindcă în Graphics Controller se mai găsește un registru cu numărul 8. În el se înscrie o mască ce indică biții care pot fi modificați în interiorul unui byte. Rolul său este foarte important și anume acela de a păstra nealterate elementele grafice anterior trasate; cu alte cuvinte, a doua coordonată denumită „poziția în interiorul byte-ului” trebuie transformată în mască. Din punct de vedere software soluția cea mai simplă este utilizarea unui tabel de măști.

Înainte de a exemplifica programul mai avem de făcut o observație. Așa cum s-a văzut în schema funcțională a cuplorului prezentată în numărul trecut, scrierea datelor de la procesor în memoria video se face printr-un ALU care trebuie alimentat cu vechia valoare printr-o operație de citire.

În limbajul C, cele două operații de citire și scriere pot fi condensate printr-un artificiu folosind operatorul „=” care nu perturbă funcția selectată în ALU.



#### LIB\_DC.C

```
unsigned char pxmask[8] =
    {0x80, 0x40, 0x20, 0x10, 0x08, 0x04,
     0x02, 0x01};

put_pixel(x,y,pixel_color)
int x,y;
char pixel_color;
{int byte;
 outp(P_ctr_a,0x01);
 outp(P_ctr_r,0xff);
 outp(P_ctr_a,0x00);
 outp(P_ctr_r,pixel_color);
 byte = (x >> 3) + RES_X*y;
 outp(P_ctr_a,8);
 outp(P_ctr_r,px_mask[x & 7]);
 * (p_ega+byte) |= 0xff;
 outp(P_ctr_a,8);
 outp(P_ctr_r,0xff);
 outp(P_ctr_a,0x01);
 outp(P_ctr_r,0x00);
 outp(P_ctr_a,0x00);
 outp(P_ctr_r,0x00);
}
```

Transformarea din sistemul de coordonate (x, y) în sistemul (byte, poziție în byte) s-a făcut prin relațiile:

ții, iar restul aceleiași împărțiri nu este altceva decât numărul cuprins în cei trei biți mai puțin semnificativi care se

```
byte int(x/8) + yx(nr_bytes/linie)
pozit = rest (x/8) sau x modulo 8
```

Deoarece 8 este egal cu 2 la puterea a treia, pentru creșterea substanțială a vitezei, împărțirea întreagă cu 8 s-a făcut prin „shiftarea” dreapta cu 3 pozi-

zilează prin mascarea cu 7.

Se mai poate observa că la terminarea operațiunii de trasare a pixelului, registrele cuplorului se readuc



într-o stare cunoscută care s-a ales ca fiind cea standard din inițializare.

Am insistat în mod deosebit asupra „problemei pixelului” fiindcă prezintă dificultăți pentru cei ce au la dispoziție doar documentația porturilor EGA/VGA și în plus este esențială pentru înțelegerea următoarelor primitive grafice. În cazul în care se dorește utilizarea VGA în modul 13H (320x200 cu 256 culori simultane) problema se banalizează fiindcă fiecărui pixel îi corespunde un byte, iar culoarea logică este dată de valoarea conținută în acesta.

Problema imediat următoare trasării unui pixel este aceea a trasării unei linii. Această problemă are două aspecte:

- clasele de funcții pentru trasarea liniei;

- algoritmul de trasarea liniei.

Există două clase mari de linii ce pot fi trasate:

- linii între coordonate absolute, la acestea precizându-se perechile de coordonate (x, y) pentru început și sfârșit. Funcția din această clasă va purta denumirea de **line()**.

- linii care se trasează între ultima coordonată a unei trasări anterioare și o pereche de coordonate (x, y) specificată. Aceasta induce necesitatea memorării coordonatelor finale ale trasării anterioare într-o pereche de variabile interne ale bibliotecii. Funcția din această clasă poartă denumirea de **draw()**. Aceasta atrage după sine funcția **move()** care are rolul de a muta originea trasării relative.

Algoritmii de trasare a unei linii pentru display-urile de tip raster nu sînt prea numeroase și vom exemplifica algoritmul Bresenham care este extrem de popular prin raportul performanțe/complexitate. Avantajul său major este că lucrează direct cu ecuația dreptei, ceea ce ar necesita operații lente de înmulțire/împărțire. El transformă totul într-un proces iterativ, bazat numai pe operații de adunare/scădere. Se avansează din pixel în pixel pe axa x sau y după un test asupra pantei și diferențialelor componentelor sale verticale și orizontale. Deoarece sursa reprezintă o extindere pentru patru cadrane, vom prezenta pentru o mai ușoară înțelegere și organizarea pentru o trasare între punctele (0,0) și (x,y).

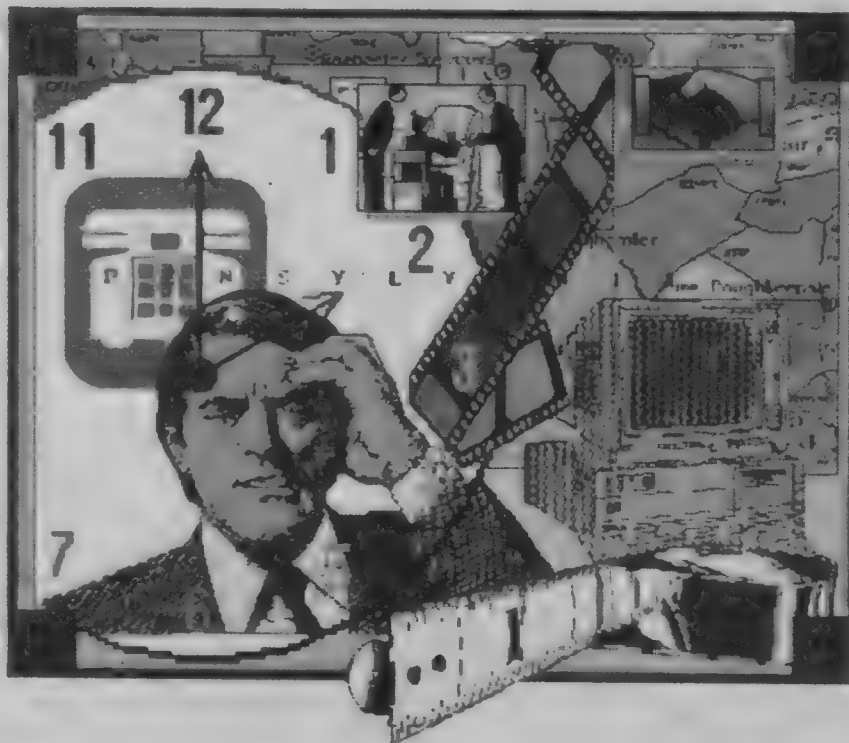
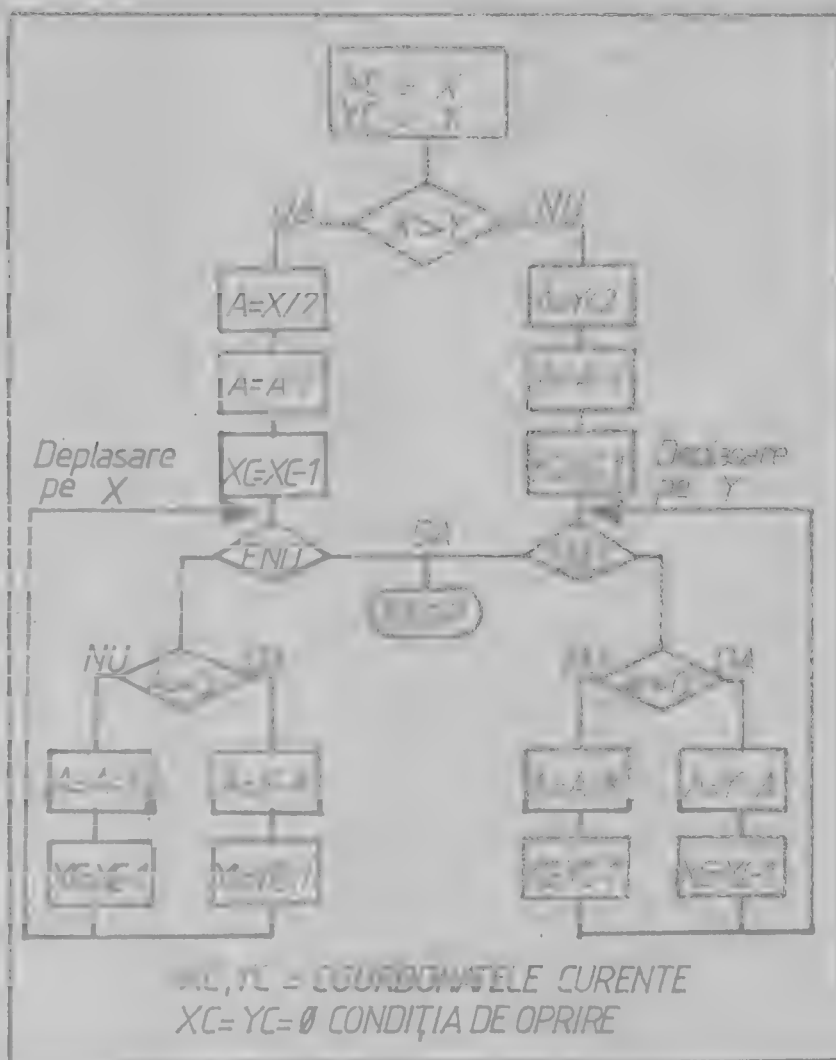
În clipa în care vom implementa acest algoritm vom observa că linia pe care o dorim noi va fi fragmentată într-o serie de segmente pe verticală sau pe orizontală, funcție de pantă. În literatura de specialitate acest fenomen poartă denumirea de **aliasing**. Există algoritmi perfecționați anti-aliasing, dar aceștia dau două tipuri de efecte secundare:

- o grosime minimă a liniei mai mare de un pixel (ex. IBM 5080);

- deși rezoluția este mare, se lucrează cu un dispozitiv virtual cu rezoluție cel puțin înjumătățită (ex. HP 9000/seria 300).

În concluzie acest efect neplăcut nu poate fi compensat decît prin rezoluție mare. La limită putem apela la 800x600, mod nestandard la VGA, dar monitorul trebuie să aibă frecvența de baleiaj pe orizontală de 40 kHz, ceea ce ne va costa minim încă 2—300 dolari.

Să analizăm acum în limbajul C:



```

unsigned int last_x,last_y;

line(x1,y1,x2,y2)
unsigned int x1,x2,y1,y2;
{outp(P_ctr_a,0x01);
 outp(P_ctr_r,0xff);
 outp(P_ctr_a,0x00);
 outp(P_ctr_r,color);
 b_line(x1,y1,x2,y2);
 last_x = x2;
 last_y = y2;
 outp(P_ctr_a,1);
 outp(P_ctr_r,0);
 outp(P_ctr_a,0);
 outp(P_ctr_r,0);
}

draw(x,y)
unsigned int x,y;
{ outp(P_ctr_a,1);
 outp(P_ctr_r,0xff);
 outp(P_ctr_a,0);
 outp(P_ctr_r,color);
 b_line(last_x,last_y,x,y);
 last_x = x;
 last_y = y;
 outp(P_ctr_a,1);
 outp(P_ctr_r,0);
 outp(P_ctr_a,0);
 outp(P_ctr_r,0);
}

move(x,y)
unsigned int x,y;
{ last_x = x;
 last_y = y;
}

```

Codul nu este scris deosebit de elegant fiindcă s-au făcut toate artificiiile necesare pentru creșterea vitezei. Cei citiva octeți pe care veți remarca că îi vom „strica” la diverse funcții pentru măști precalculate sau scoaterea unor apeluri de funcții în afara buclelor, vor aduce un „profit” deloc neglijabil sub aspectul vitezei.

Deși există riscul să vă pierdeți răbdarea, nu putem încă părăsi domeniul banalei linii deoarece ea mai are încă

atribute specifice.

Unul din ele este pattern-ul cu care se trasează linia, deci o mască care să acopere ciclic pixelii. Soluția este banală și constă într-un AND al măștii registrului 8 al P-ctr cu biții succesivi ai pattern-ului. Deoarece în limbajul C nu avem operații de rotație (ci numai de shiftare) vom apela la un artificiu și anume refolosirea setului de măști prestabilite, **pxmask**, cu indexul preluat după un contor. Nu este necesar să impunem un nou contor, el există,

```

b_line(xp,yp,xf,yf)
unsigned xp,yp,xf,yf;
{register int iv,px,py;
 int byte,nx,ny,nt,nd,ns;
 unsigned int anx,any;
 nx = xf-xp;
 ny = yf-yp;
 px = (nx > 0) ? 1: -1;
 py = (ny > 0) ? 1: -1;
 anx = abs (nx);
 any = abs (ny);
 if(anx > any)
 {nt = anx;
 nd = any;}
 else
 {nt = any;
 nd = anx;}
 ns = nt - nd;
 iv = (nt >> 1) - ns;
 while(nt)
 {byte = (xp>>3) + RES_X*yp;
 outp = (P_ctr_a,8);
 outp = (P_ctr_r,pxmask[xp & 7]);
 *(p_ega+byte) |= 0xff;
 nt--;
 if(iv >= 0)
 {iv -= ns;
 xp += px;
 yp += py;}
 else
 {iv += nd;
 if(anx > any)
 xp += px;
 else
 yp += py;
 }
 outp = (P_ctr_a,8);
 outp = (P_ctr_r,0xff);}
}

```

și anume variabila nt.

Atenție! Înainte de a modifica funcția **b\_line()** (Bresenham line) faceți cu editorul o copie numită **e\_line()** (echo line). Această nouă funcție ne va fi necesară la trasarea unor cursoare grafice, care trebuie să acționeze în timp real, nu au atribut de pattern și trebuie să poată fi trasate cu maxim de viteză.

Atributul de pattern de linie, va fi o variabilă internă bibliotecii și va fi introdus cu o funcție specifică.

(continuare în pag.27)

```

LIB_BAS.C
unsigned char style = 0xff;

line_style(pattern)
unsigned char pattern;
{style = pattern;
}

```

Impactul microcalculatoarelor personale compatibile IBM a fost foarte mare și datorită facilităților grafice pe care le oferă acestea în diferite opțiuni.

Iată câteva dintre ele:

- CGA — Color Graphic Adapter (definiție 640x240 pixeli);
- PGC — Professional Graphic Controller;
- MDA — Monochrome Display/Printer Adapter (80 semne, 25 rânduri, matrice 8x14 pixeli);
- MGC — Monochrome Graphic/Printer Card
- HGC — Hercules Graphic Card; producător Hercules Computer Technology.

Hercules poate lucra în două moduri: ● mod grafic: definiție 720x348 pixeli ● mod text: 80x25, matrice 9x14 pixeli

— frecvență de baterie: ● orizontală 18,52 kHz ● verticală 50 Hz

Denumirea de Hercules este marcă înregistrată a firmei Hercules Computer Technology. De multe ori versiuni echivalente pot avea alte denumiri cum ar fi, de pildă, MGA (Monochrome Graphics Adapter) și altele, deoarece denumirea de „Hercules” este protejată.

Placa HGC — în varianta a II-a este executată cu circuite integrate de tip MSI (medium scale of integration), iar următoarele versiuni, mai ales HGC IV au câștigat mult în densitate prin folosirea circuitelor specializate de tip LSI. Descrierea ultimei versiuni (cea de-a patra) ar fi foarte laborioasă și dificilă, motiv pentru care, în cele ce urmează, vom face o prezentare mai amplă a versiunii a II-a.

**Modul text.** Imaginea este plasată în memorie în zona 0B0000H-0B0FFFH (4 KB). Fiecare semn este însoțit de 2 bytes, reprezentând codul ASCII și respectiv atributele. Codurile de definire a semnelor sînt memorate secvențial cu 160 bytes pe rând.

Adresa unui semn plasat în coloana X (în intervalul 1—80), și rândul Y (în intervalul 1—25), față de adresa de bază se poate calcula astfel:

$$160x(Y-1) + 2x(X-1)$$

Generatorul de semne permite obținerea de 256 simboluri (specificate de codurile Ascii) diferite. A doua diferențiere — atributele care sînt următoarele: underline, blank, bold face (deschis la culoare), blink sau reverse video.

**Modul grafic** se caracterizează prin faptul că fiecare bit reprezintă un punct de pe ecran. Schimbarea conținutului punctului (X, Y) cu  $X \in 0,719$  și  $Y \in 0,347$ , modificat față de adresa de bază a paginii 0-0B0000H sau 0B8000H pentru pagina 1 se calculează astfel:

$$2000 Hx(Y \text{ mod } 4) + 90 x INT(Y/4) + INT(X/8)$$

Locul bitului în byte-ul care conține punctul (X,Y) este:  $7-(X \text{ mod } 8)$ .

Cea mai comodă este folosirea procedurilor MS-DOS, mai ales în modul text. În modul grafic, HERCULES a

lansat o dată cu opțiunea HGC și programele HBASIC și Graph X. În acest ultim pachet se regăsesc 15 proceduri grafice elementare (primitives) operate în mod mașină sau într-un limbaj de nivel superior. Programele LOTUS 1-2-3, Turbo Pascal, Auto CAD sînt dotate cu proceduri proprii de comandă a imaginii grafice (drivers) și cu algoritmi de desenare rapidă.

**Modul de codificare a atributelor.** Poziția bitului:

7	6	5	4	3	2	1	0	Atribut
B	0	0	0	1	0	0	0	(blank)
B	0	0	0	1	0	0	0	(underline)
B	0	0	0	1	1	1	1	(normal video)
B	1	1	1	1	0	0	0	(reverse video)

1 = 1 deschis la culoare

B = 1 dacă  $b_5 = 0$  fond deschis și  $b_5 = 1$  pîlpiire

**Comanda display-ului.**

Comanda monitorului monocromatic se face cu ajutorul a 5 registre (la adresele indicate în paranteze):

- Index Register (03B4H);
- Data Register (03B5H);
- Display Mode Control Port (03B8H);
- Display Status Port (03BAH);
- Configuration Switch (03BFH);



# HERCULES II 1

**1** BLOC  
MEMORIE  
IMAGINE

**2** BLOC  
FORMARE  
IMAGINE

**3** BLOC  
PARAMETRI  
IMAGINE

**4** MEMORIE  
ATRIBUTE  
IMAGINE

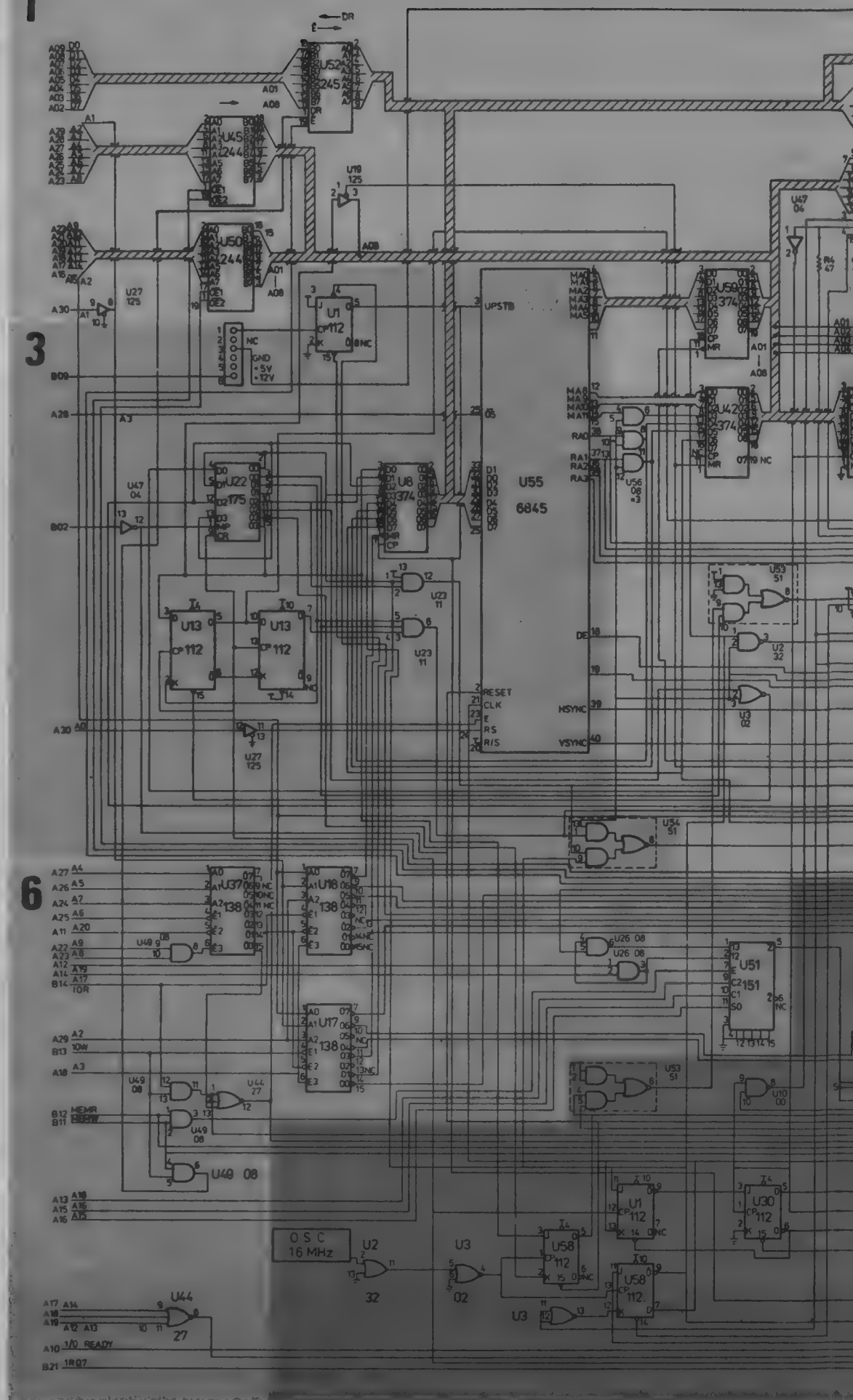
**5** GENERATOR  
SEMNE

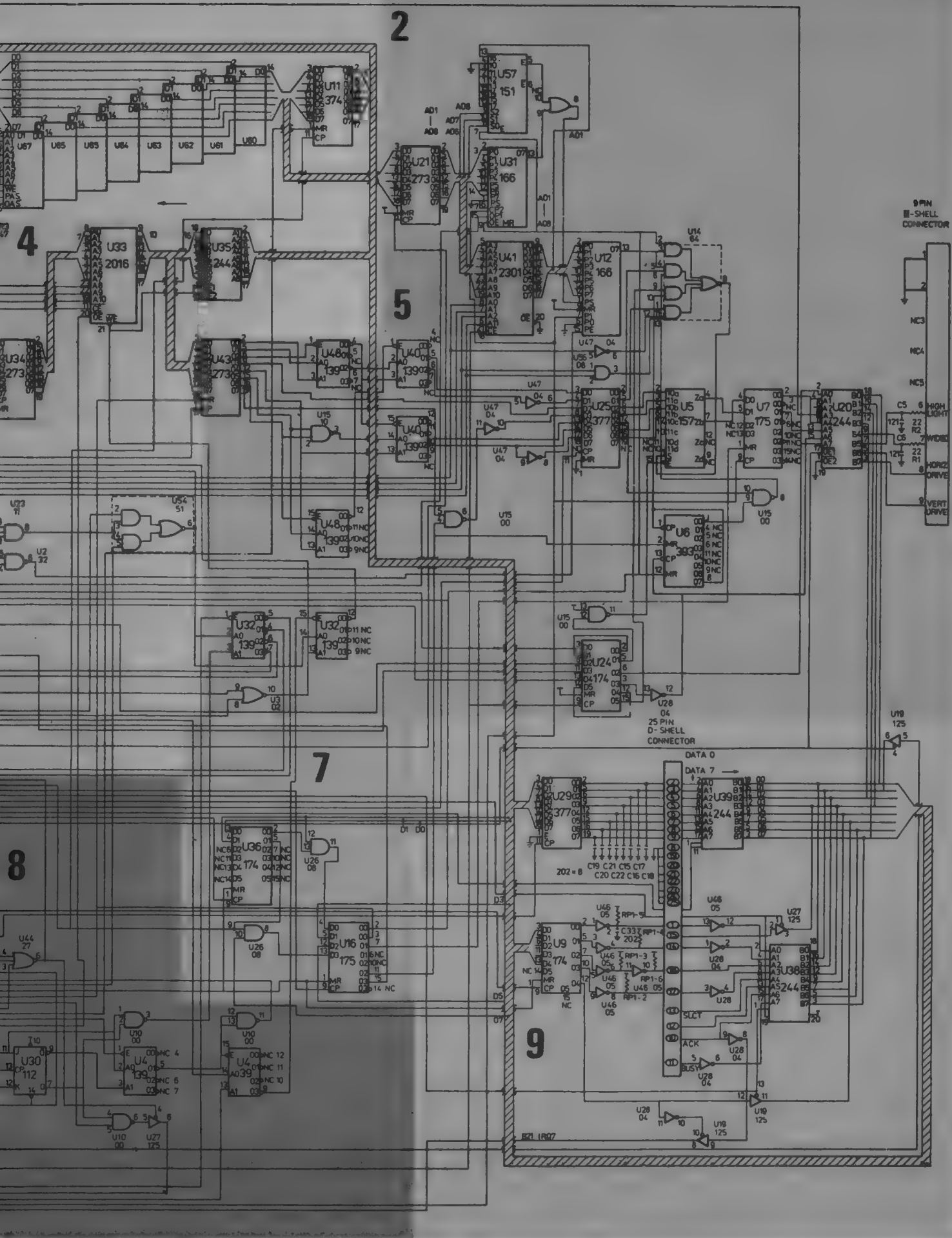
**6** BLOC  
DECODIFICARE  
ADRESE  
PLACH.

**7** REGISTRE  
COMANDA  
ILUMINARE

**8** BLOC  
COMANDA  
ACCES MEMO-  
RIE IMAGINE

**9** BLOC  
CONECTOR  
IMPRIMANTA





## Conector imprimantă

Pin	Polarizare — registru — direcție	Obs.
1	— STROBE/OUT	impuls selecție date
2-9	D0-D7/OUT	date
10	— ACK/IN	impuls confirmare primire date
11	+ BUSY/IN	echipament ocupat
12	+ END PAPER/IN	sfârșit hirtie
13	+ SELECT STATUS/IN	imprimare-ready
14	— AUTO FEED/OUT	comandă hirtie
15	— ERROR/IN	eroare imprimare
16	— INIT/OUT	inițiere imprimantă
17	+ SELECT IN/OUT	activare imprimantă
18-25	GROUND	masă

## Registru display

(b<sub>0</sub>, b<sub>2</sub>, b<sub>4</sub>, b<sub>6</sub> — neutilizate)

a) mod	bit	opțiune
+ SELECT GRAPHIC MODE	b <sub>1</sub>	0 — mod text 1 — mod grafic 1 — deconectare imagine
+ ENABLE CHAR BLINK	b <sub>5</sub>	0 — deconectare 1 — conectare pilpiire
+ PACE NO.	b <sub>7</sub>	0 — pagina 0 1 — pagina 1
+ ENABLE VIDEO OUTPUT	b <sub>3</sub>	0 — întunecare ecran
b) stare		
+ HORIZONTAL SYNC	b <sub>0</sub>	0 — semne normale 1 — sincronizare orizontală, ecran întunecat
+ VIDEO OUTPUT	b <sub>3</sub>	0 — ecran neactiv
— VERTICAL RETRAGE	b <sub>7</sub>	0 — retur spot, ecran întunecat 1 — afișare activă

## Registru configurație

Bit	opțiune
b <sub>0</sub>	0 — asigurare împotriva conectării mod grafic 1 — permite conectarea mod grafic
b <sub>1</sub>	0 — interzicere alegere pagina 1 1 — permite alegere pagina 1

Conținutul imaginii provine din memoria DRAM (U60 - 67) de câte 64 kb dinamică și U33 - 2kB statică, pentru 2-900 attribute.

## Interfața imprimantă

Pe placheta HGC se află interfața CENTRONICS paralelă pentru imprimantă. Comanda se face prin 3 registre in-out: PRINTER DATA PORT - 3BCH, PRINTER CONTROL PORT - 03BDH.

## Comanda imprimantă

(b<sub>5</sub>, b<sub>6</sub>, b<sub>7</sub> — nefolosiți)

+ STROBE	b <sub>0</sub>	0 — impuls de scriere 1 — stingere impuls (= deconectare alimentare)
+ AUTO FEED 1	b <sub>1</sub>	0 — avans automat hirtie 1 — avans comandat de la distanță
— INIȚIALIZARE	b <sub>2</sub>	0 — inițiere imprimantă 1 — funcționare normală
+ SELECT	b <sub>3</sub>	0 — deconectare logică imprimantă 1 — conectare
	b <sub>4</sub>	0 — mascare IRQ 7 1 — deconectare IRQ 7 (imprimanta gata de recepționat date)

## Stare imprimantă

(b<sub>0</sub>, b<sub>1</sub>, b<sub>2</sub> neutilizați)

— ERROR	b <sub>3</sub>	0 — greșală imprimantă 1 — funcționare corectă
+ SELECT STATUS	b <sub>4</sub>	0 — deconectare imprimantă 1 — conectare imprimantă
+ PAPER OUT	b <sub>5</sub>	0 — funcționare normală 1 — lipsă hirtie
— ACK	b <sub>6</sub>	0 — imprimanta prelucurează date 1 — imprimanta poate primi date
— BUSY	b <sub>7</sub>	0 — imprimanta ocupată (neactivă, defectă) 1 — imprimanta ready

## Registru index

Index	Descriere	Mod grafic	Mod text
0	Rînd complet, inclusiv sincronizare	35 H	61 H
1	Număr semne vizibile pe ecran	3 DH	50 H
2	Plasare semn de inițializare sincro. oriz.	2 EF	52 H
3	Nr. semne în impuls sincro. oriz.	07 H	0 FH
4	Nr. total rînduri în imagine	5 BH	19 H
5	Nr. linii +4 pentru obținerea a 50 imagini/secundă	02 H	19 H
6	Număr rînduri vizibile	57 H	19 H
7	Număr rînduri început sincronizare Verticală (impuls sincro durează peste 16 linii orizontale)	57 H	19 H
8	Mod afișare (0 sau 2 imagine normală)	02 H	02 H
9	Număr linii în rînd	03 H	0 DH
10	Linia din rînd unde cursorul este peste semn	00 H	08 H
11	Linia din rînd de la care cursorul nu se mai suprapune peste semn	00 H	0 CH
12, 13	Numărul semnului de la care începe afișarea	00 H	00 H

### Atenție!

Experimentarea cu parametrii imaginii fără cunoașterea exactă a descrierii tehnice a circuitului CRTC 6845 - MOTOROLA poate duce la distrugerea monitorului.



```
LIB.H:
#define COPY 0
#define AND 1
#define OR 2
#define XOR 3
```

```
LIB_BAS.C
unsigned char replacement_rule;

drawing_mode(mode)
    unsigned char mode;
{replacement_rule = (mode<<3) &0x18;
  outp(P_ctr_a,3);
  outp(P_ctr_r,replacement_rule);
}
```

Din pacate pentru diverse pattern-uri vom constata cit de neplăcute sînt efectele aliasing-ului. Mai putem compensa cite ceva trecînd la un pattern pe 16 biți, dar complicarea programului nu justifică îmbunătățirea.

Alt atribut este grosimea liniei. Grosimea constă în trasări succesive cu alterarea cu cite un pixel a coordonatelor liniei. Această alterare se face pe baza valorilor rotunjite ale funcțiilor sin și cos calculate pentru panta drepte. Deoarece nu reprezintă o problemă de algoritm grafic sau cuplul EGA/VGA deosebită, vă lăsăm plăcerea să implementați singuri funcția `line-width()` în fișierul LIB-BAS.C și să alternați corespunzător funcția `line()` din fișierul LIB-DC.C.

Un ultim atribut, dar deosebit de spectaculos este modul de trasare. Revăzînd schema funcțională a lui EGA/VGA din numărul trecut, vom observa că scrierea unui nou pixel nu se face direct, ci printr-un ALU cu patru funcții. Deoarece cursoarele grafice și alte funcții lucrează cu unele moduri bine precizate, este necesară ca și în alte cazuri anterioare, memorarea stării într-o variabilă internă bibliotecii pentru restaurarea stării cuplului după utilizarea primitivelor menționate. Pentru a ușura scrierea și citirea unor programe de aplicații, valorile pentru comanda funcțiilor ALU vor fi stocate în header-ul bibliotecii sub forma unor constante simbolice.

## Dialog cu cititorii

Ne propunem, ca în cadrul acestei rubrici să avem un dialog deschis cu cititorii asupra unor multiple probleme cu care se confruntă informatica în țara noastră. Pe de altă parte, tot în paginile revistei, sub antetul „COMPUTER-WORLD” răspundem la o mică parte din întrebările care ne-au fost adresate cu privire la noutățile din domeniu folosind bibliografia IDG pe care o primim cu regularitate la redacție.

Revenind la rubrica „SEMNAL”, ne-am hotărît, cu acordul autorului, să publicăm un fragment dintr-o scrisoare pe care am primit-o la redacție și care ne-a creat o mare tristețe. Autorul, Mathe Stefan, este animatorul și conducătorul clubului „Spectrum” din Tirgu Mureș. Cele spuse de Stefan Mathe nu necesită comentarii. Ele sînt de prisos. Noi sperăm, însă, că aceste rînduri vor fi citite și de cei care ar trebui să...

„Pot să vă anunț că Clubul SPECTRUM din Tirgu Mureș există în continuare, în ciuda tuturor greutăților ce s-au ivit. Nu mă pot lăuda că avem o activitate foarte bogată, dar măcar odată pe săptămînă ne întîlnim, facem schimburi de programe, documentații și, mai ales, de experiență. Aceste întîlniri au în centrul lor de cele mai multe ori o prezentare organizată a unui software utilitar, urmînd discuții libere. Unul dintre cîștigurile acestor întruniri este că elevii care vin acolo au ocazia să butoneze puțin la cele cinci calculatoare HC-85 care ne-au rămas prin bunăvoința primăriei municipiului, după ce o parte din întreprinderi și-au retras calculatoarele din sala noastră. În altă zi din săptămînă desfășurăm cursuri de inițiere în programarea acestor calculatoare. Începînd cu acest an, inițierea o facem doar cu adulți, deoarece considerăm că este mai eficient așa, ei pot transmite mai multor copii, la rîndul lor, cunoștințele acumulate. După cum am putut aprecia, cursul nostru este mai complex decît cel susținut de Universitatea Populară (sau cum s-o mai fi numind acum) ori de Casa Tineretului din municipiu. Nemaivorbind că celelalte cursuri se fac contra unei taxe de 500 lei. În altă zi, cursanții pot veni să lucreze individual, fără profesor.

Consider că avem cele mai bune condiții pe care, la nivelul actual din dotarea învățămîntului cu calculatoare și-l poate imagina cineva.

Avem însă două motive majore de amărăciune:

- profesorii nu doresc cu nici un chip să învețe, deși este mare nevoie de cadre care să predea această disciplină (care are un capitol aparte și în manualele de clasa a VII-a și VIII-a - nu punem acum în discuție nivelul acestui capitol);

- nici o școală nu dorește să ne asociem, să umplem sala cu calculatoare pentru a putea lucra cu un număr mai mare de cursanți odată, ei beneficiînd în aceeași măsură în care beneficiem noi. Calculatoarele care sînt în prezent în dotarea școlilor sînt, în multe cazuri, ținute acasă de unii profesori.”

Încheiem aici citatul. În continuare autorul ei se referă cu spirit critic la ceea ce a găsit și la ceea ce nu a găsit în paginile primului număr al „INFO-CLUB”. Nu publicăm restul din lipsă de spațiu tipografic, dar îl mulțumim și

răspundem pe aceasta cale, tuturor celor care ne-au întrebat, că INFOCLUB se poate obține direct de la redacție, prin trimiteră mandatului poștal cu contravaloarea de 1x35 sau 4x35 lei (anul acesta apar 4 numere, iar de la

anul lunar) pe adresa: GHEORGHE BADEA, Piața „Presă Liberă” nr. 1, București, Cod 79 781, pentru revista „INFOCLUB”.

Vom reveni în numerele viitoare cu diferite opinii privind soarta informaticii în România.

Acesta este sloganul publicitar cu care Big Blue își anunță revenirea în fruntea plutonului PC prin noile sale mașini IBM PS/2 Model 90 XP 486 și Model 95 XP 486.

Să vedem mai întâi care este componența plutonului. Pe undeva pe la coadă, mai aleargă încă mașinile XT. Pentru ele compu-gerontologia a lucrat din plin: procesoare NEC V.2x în 10 și 12 MHz, cuploare VGA cu monitoare monocrome și color, discuri fixe cu capacitate mai mare și timp de acces mai mic. Urmează apoi un grup important bazat pe microprocesorul 286 lucrând în 14/16/20 MHz. Declinul lor este evident și marile firme păstrează în nomenclatorul lor doar unul sau două modele pentru aplicațiile nepretențioase și clienți cu bani puțini. De fapt în ultima vreme pentru aceștia au devenit mult mai accesibile și mai tentante mașinile bazate pe microprocesorul 386 SX lucrând la 16/20 MHz. Ajungem în sfârșit la mașina caracteristică anului 1990 care dispune de un procesor 386 lucrând la 25/33 MHz, memorie standard de 2 MB, memorie cache internă de 32 KB, adaptor grafic Super VGA (1024x768) și discuri Winchester de cel puțin 80 MB. În aceste condiții anul 1991 va fi oare anul calculatoarelor bazate pe microprocesorul 486? Probabil că da!

În toată această evoluție nu este vorba numai de creșterea frecvenței ceasului microprocesorului și a dimensiunii bus-ului de date ci și de multe alte îmbunătățiri calitative care sporesc sensibil puterea de lucru. Iată sintetic, într-un tabel furnizat de firma Intel, evoluția procesoarelor sale:

Să revenim însă la cele două noi modele ale firmei IBM anunțate la început. Ele au o arhitectură internă sofisticată care caută să exploateze întregul potențial al microprocesorului Intel 486. Denumirea Expandable Processor (XP) evidențiază posibilitatea adăugirilor și îmbunătățirilor ulterioare care să prelungească timpul de serviciu al unui astfel de sistem într-un domeniu în care evoluțiile tehnologice se petrec cu o viteză surprinzătoare.

În ceea ce privește bus-ul intern s-a optat tot pentru Micro Channel (MCA) ca și la alți membri ai familiei PS/2 dar într-o versiune de 32 de biți și cu o viteză de transfer îmbunătățită, versiune care permite cu ajutorul cuploarelor de tip „master” să se adauge până la alte 16

procesoare. Deși această facilitate există, Bob Carberry — asistent al managerului general pentru tehnologia sistemelor din departamentul de sisteme personale, a anunțat că deocamdată pentru IBM un singur procesor este de-ajuns deoarece Intel este capabilă să dubleze performanțele microprocesoarelor la fiecare 18 luni.

Cuplorul de disc este în mod surprinzător conform standardului industrial SCSI dar cu perfecționări pentru a putea suporta o viteză de transfer foarte mare. Tot în sensul creșterii ratei transferului se poate opta și pentru un cache de 256 kB! Pentru lucrul în rețea cu baze de date foarte mari există două opțiuni cu ajutorul cărora se pot gestiona resurse enorme:

	Model 90	Model 95
Hard Disk Expansion Bay	0,36 GB	1,6 GB
External Storage Enclosures	5,16 GB	8,36 GB

Să nu neglijăm însă adaptorul grafic care introduce un nou standard: Extended Graphics Array (XGA). Folosind un monitor performant tip IBM 8512 avem pe lângă modurile de lucru VGA tradiționale unele noi, deosebit de performante:

● Moduri grafice cu organizare pe plane: 640x480 — 256 culori/64

gradații de gri; 1024x768 — 16 culori/16 gradații de gri;

● Moduri grafice cu acces direct (pe 16 biți): 640x480 — 64 K culori;

● Moduri alfanumerice: 50x132.

În final vă prezentăm un tabel cu configurațiile celor două modele:

	Model 90	Model 95
MICROPROCESSOR		
Standard	80486	80486
Frecvența ceas	25—33 MHz	25—33 MHz
MEMORIE		
Standard	4 MB (70 ns)	4 MB (70 ns)
Expand	32 MB	16 MB
ADAPTOR DE		
INTRODUCERE	Extended Graphics Array (XGA)	Extended Graphics Array (XGA)
	Port de adaptor DMA	Port de adaptor DMA
	1	1
	Port pentru adaptor DMA	Port pentru adaptor DMA
	Controler de disc hard tip 1, 3 unități	Controler de disc hard tip 1, 3 unități
	Controler de disc fix (SCSI) cu cache	Controler de disc fix (SCSI) cu cache
DISC FIX		
Standard	0,36—0,72 MB	1,6—8,36 MB
SLOT-URI LIBRE	3 x 32 biți	6 x 32 biți
BUS	MCA 32 biți	MCA 32 biți

## Borland umple ferestrele goale cu Visual Apps Generator

INFOCLUB 7 ianuarie 1991

Intrarea pe piața produselor Windows anunțată cu mult timp în urmă este așteptată către mijlocul anului sub forma unui generator de aplicații care utilizează tehnicile de programare vizuală în scopul de a ajuta utilizatorii să creeze aplicații fără a utiliza un limbaj de programare.

Specialiștii de la firma Borland au confirmat că programul bazat

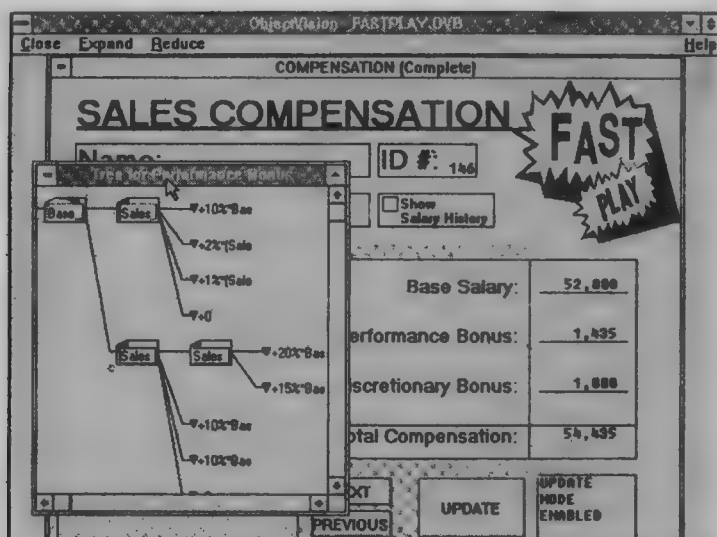
pe Windows, ce poartă denumirea de Object Vision, permite utilizatorilor crearea aplicațiilor prin construirea unui formular pe ecran și asignarea caracteristicilor și comportamentului, fiecărui element al formularului. Aceste caracteristici sînt prezentate sub forma unui arbore decizional cu trasee care ajută utilizatorul să „navigheze” în interiorul formularului.

celelalte din tabele. Din surse ale companiei, s-a afirmat că Borland situează Object Vision ca un superset al unui pachet spreadsheet.

Surse autorizate din mediul industrial afirmă că Borland va ridica nivelul lui Object Vision deasupra masei actuale de utilizatori. „Borland este deja în frunte pe această direcție de multă vreme și are deja un număr mare de clienți care au realizat aplicații cu Quattro, Paradox și limbajele bazate pe ele”, a afirmat Tarter.

Compania subliniază, de asemenea, diferența între Object Vision și integratorul de date denumit Info Alliance de la Software Publishing Corp, despre care Borland afirmă că nu poate fi considerat un generator de aplicații pentru utilizatorul curent. De altfel, Info Alliance pornește de la 8 500 \$ pentru 10 utilizatori, pe cînd Object Vision va costa „prețul mediu al unui produs spreadsheet”, au afirmat oficialii ai firmei Borland.

**Traducere și adaptare  
de Eugen GEORGESCU**



Formularele Object Vision pot reprezenta formulare reale, ca de exemplu formulare medicale, dar sînt, de asemenea, și o formă simbolică pentru stocarea și accesul datelor — foarte asemănător cu fișele și stivele de la Hypercard (de la Apple). Chiar dacă Object Vision are aceleași caracteristici funcționale ca și alte produse similare, programul se situează deasupra acestor capacități prin interconectarea unor facilități de spreadsheet și gestiunea bazelor de date.

Firma din Scotts Valley, California, și-a remodelat în mod aparent produsul de la debutul liniștit la închiderea Comdex-ului, cînd el era numit Expresso și descris ca un produs pentru formularistică. Analisti familiarizați cu Object Vision au afirmat că legătura sa strînsă cu alte aplicații îl situează cu o clasă

deasupra procesoarelor de formularistică (spreadsheet). „Este un sistem pentru construit utilitare”, a afirmat Jeff Tarter, publicist la Softletter, o publicație de „scandal” din mediul industrial.

Object Vision, care poate accesa datele în mod concurent din surse multiple ce se află în rețea, poate lucra cu Dbase, Btrieve, Paradox, fișiere ASCII cu cîmpurile delimitate prin virgulă și aplicații care suportă schimburile dinamice de date sub Windows. Legătura cu raportările bazate pe limbajul SQL vor fi implementate mai tîrziu.

Logica lucrului cu Object Vision este foarte asemănătoare cu logica declarativă din alte produse pentru spreadsheet. Utilizatorul poate defini cîmpuri, fie în interiorul unui formular, fie în interiorul unei baze de date de o manieră similară cu



Ion Diamandi,  
Florin Vasiliță

# RUTINĂ GRAFICĂ pentru UMPLERA UNOR CONTURURI

Răspundem prin acest articol numeroaselor scrisori și solicitări din partea utilizatorilor de calculatoare personale, în speranța că, în acest fel, vom satisface nevoile mult mai multor posesori de astfel de calculatoare care, din diferite motive, nu au reușit să ne scrie până în acest moment. Prezentăm deci, o rutină foarte performantă realizată în cod mașină și utilizabilă în programe pentru umplerea rapidă a unor contururi.

## Principiul de realizare a rutinei și descriere.

Se caută începutul liniei de pixeli în care se află punctul curent analizat. De aici (de la început) se umple linia până la găsirea unui punct activat sau a sfârșitului (marginii) de ecran. Dacă se găsește o ramificație, aceasta este depusă în lista FIFO. Dacă în listă se găsește o ramificație, atunci se aplică umplerea de linie și se scoate din listă. Noua ramificație se depune la sfârșitul listei iar citirea se face de la început.

În locul coordonatelor punctelor se utilizează adresa + mască. În mască un bit setat va da un punct. Lista FIFO are 400 de locații (o locație = 3 octeți). WRITER indică următoarea locație liberă, iar READER, ramificația cea mai veche.

PUTTUB depune o ramificație, iar GETTUB extrage o ramificație. Umplerea unui rând se realizează cu subrutina PLINE. Prin intermediul subrutinelor HLDOWN și HLUP se solicită o adresă din memoria RAM pentru ecran și se furnizează adresa octetului care este „sub” și, respectiv, „deasupra” adresei. Dacă lista FIFO este plină PUTTUB returnează Z = 1.

**Performanțe:** rutina poate umple până la maximum 10 000 puncte (pixeli) pe secundă (de exemplu, un patrat de 100 x 100 este umplut în circa o secundă).

## Rutina de umplere în cod mașină pentru calculatoare HC:

```
10      ORG  #F000
12
14 PAINT:
16      PUSH AF
18      PUSH BC
20      PUSH DE
22      PUSH HL
24      CALL TUBINI
26      CALL PIXADD
28      CALL MASKG
```

```
30      CALL MODE
32      CALL PUTTUB
34
36 PAINT1:
38      CALL PLINE
40      JR NZ.PAINT1
42      POP HL
44      POP DE
46      POP BC
48      POP AF
50      RET
52
54 MODE:
56      LD C,0
58      CALL POINT
60      RES 3,C
62      RET Z
64      SET 3,C
66      RET
68
70 TUBINI:
72      LD HL,FIFO
74      LD (READER),HL
76      LD (WRITER),HL
78      LD DE,(WRITER)
80      RET
82
84 GETTUB:
86      LD HL,(READER)
88      LD DE,(WRITER)
90      CALL CP16
92      RET Z
94      INC HL
96      LD E,(HL)
98      INC HL
100     LD D,(HL)
102     INC HL
104     LD B,(HL)
106     PUSH DE
108     LD DE,FIFOE
110     CALL CP16
112     JR NZ.GET1
114     LD HL,FIFO
116
118 GET1:
120     LD (READER),HL
122     POP HL
124     XOR A
126     INC A
128     RET
```

```
130 PUTTUB:
132     EX DE,HL
134     PUSH
136     LD HL,(WRITER)
138     INC HL
140     LD (HL),E
142     INC HL
144     LD (HL),D
146     INC HL
148     LD (HL),B
150     LD DE,FIFOE
152     CALL CP16
154     JR NZ.PUT1
156     LD HL,FIFO
158
160 PUT1:
162     LD DE,(READER)
164     CALL CP16
166     JR Z.PUT2
168     LD (WRITER),HL
170
172 PUT2:
174     POP HL
176     RET
178
180 PLINE:
182     CALL GETTUB
184     RET
186     CALL SLINE
188     SET 0,C
190     SET 1,C
192
194 PLINE1:
196     LD A,(HL)
198     CALL POINT8
200     JR Z.PLINE2
202
204 PLINE3:
206     CALL POINT
208     RET
210     CALL PLOT
212     CALL BOT1
214     CALL TOP1
216     RRC B
218     JR NC.PLINE3
220     JR PLINE5
222
224 PLINE2:
226     XOR
228     BIT 3,C
230     JR NZ.PLINE4
```

```

232      CPL
234
236 PLINE4:
238      LD      (HL).A
240      CALL TOP8
242      CALL BOT8
244
246 PLINE5:
248      INC     HL
250      LD      A.L
252      AND     #1F
254      JR      NZ.PLINE1
256      INC     A
258      RET
260
262 POINT:
264      LD      A.(HL)
266      AND     #
268      BIT     3.C
270      JR      NZ.PO1
272      OR      #
274      RET
276
278 PO1:
280      XOR     B
282      RET
284
286 PLOT:
288      LD      A.(HL)
290      OR      B
292      BIT     3.C
294      JR      Z.PL1
296      XOR     B
298
300 PL1:
302      LD      (HL).A
304      RET
306
308 BOT1:
310      PUSH    HL
312      CALL HLDOWN
314      LD      DE.SCREEN
316      CALL CP16
318      JR      NC.BOT12
320      CALL BOTB
322
324 BOT12:
326      POP     HL
328      RET
330
332 BOTB:
334      CALL POINT
336      JR      Z.BOTB1
338      SET     0.C
340      RET
342
344 BOTB1:
346      BIT     0.C
348      RET     #
350      CALL PUTTUB
352      RES     0.C
354      RET
356
358 TOP1:
360      PUSH    HL
362      CALL HLVP
364      LD      DE.SCREEN
366      CALL CP16
368      JR      C.TOP12
370      CALL TOPB
372
374 TOP12:
376      POP     HL
378      RET
380
382 TOPB:
384      CALL POINT
386      JR      Z.TOPB1
388      SET     1.C
390      RET
392
394 TOPB1:
396      BIT     1.C
398      RET     #
400      CALL PUTTUB
402      RES     1.C
404      RET

```

```

406
408 HLDOWN:
410      INC     #
412      LD      A.H
414      AND     7
416      RET     NZ
418      PUSH    #
420      LD      DE.-#7E0
422      ADD     HL.DE
424      POP     DE
426      LD      A.H
428      AND     7
430      RET     Z
432      LD      A.H
434      ADD     A.7
436      LD      H.A
438      RET
440
442 HLVP:
444      LD      A.H
446      AND     7
448      JR      Z.HLVP1
450      DEC     #
452      RET
454
456 HLVP1:
458      LD      A.L
460      AND     #E0
462      JR      Z.HLVP2
464      PUSH    #
466      LD      DE.#6E0
468      ADD     HL.DE
470      POP     #
472      RET
474
476 HLVP2:
478      LD      A.L
480      SUB     #20
482      LD      L.A
484      DEC     H
486      RET
488
490 SLINE:
492      CALL LEFTB
494      JR      C.SLINE1
496      RRC     B
498      RET
500
502 SLINE1:
504      # B
506
508 SLINE2:
510      LD      A.L
512      AND     #1F
514      RET     Z
516      # HL
518      LD      A.(HL)
520      CALL POINTB
522      # Z.SLINE2
524      RLC     #
526      CALL LEFTB1
528      RRC     #
530      RET     NC
532      INC     HL
534      RET
536
538 LEFTB:
540      RLC     B
542      RET     C
544
546 LEFTB1:
548      CALL POINT
550      # Z.LEFTB
552      # #
554      RET
556
558 CP16:
560      LD      A.H
562      CP      #
564      RET     NZ
566      LD      A.L
568      CP      #
570      RET
572
574 BOT8:
576      PUSH    HL
578      CALL HLDOWN

```

```

580      LD      DE.SCREEN
582      CALL CP16
584      JR      NC.BOT8N
586      LD      A.(HL)
588      OR      #
590      JR      Z.BOT81
592      CPL
594      JR      NZ.BOT82
596
598 BOT81:
600      CALL #
602      POP     HL
604      RET
606
608 BOT82:
610      CALL #
612      RRC     #
614      JR      NC.BOT82
616
618 BOT8N:
620      POP     HL
622      RET
624
626 TOP8:
628      PUSH    HL
630      CALL HLVP
632      LD      DE.SCREEN
634      CALL CP16
636      JR      C.TOP8N
638      LD      A.(HL)
640      OR      #
642      JR      Z.TOP81
644      CPL
646      JR      NZ.TOP82
648
650 TOP81:
652      CALL TOPB
654      POP     HL
656      RET
658
660 TOP82:
662      CALL TOPB
664      RRC     B
666      JP      NC.TOP82
668
670 TOP8N:
672      POP     HL
674      RET
676
678 POINT8:
680      BIT     3.C
682      JR      Z.PO81
684      CPL
686
688 PO81:
690      OR      A
692      RET
694
696 MASKG:
698      LD      B.1
700      INC     A
702
704 MG1:
706      RRC     B
708      DEC     A
710      JR      NZ.MG1
712      RET
714
716
718 FIFO      EQU     #
720      DEFS    400*3
722 FIFOE      EQU     #-1
724
726 READER     DEFS    2
728 WRITER     DEFS    2
730
732 PIXADD     EQU     #22AA :
734 SCREEN     EQU     #4000
736 SCREND     EQU     #57FF
738
740 TEST       LD      BC.(#3C7D) :
742 ULTIMUL     PUNCT
744      JP      PAINT
746      END     TEST

```

(continuaré en pag.47)

CHILD

UTILIZATION

Thinking about your help





# SISTEMUL DE OPERARE MS-DOS PENTRU CALCULATOARELE PERSONALE

## Introducere

Un sistem de operare este un ansamblu de programe care realizează legătura, comunicarea, dintre hardware, utilizatorul uman și alte sisteme software.

Sistemele de operare sînt strîns legate de mașină, de aceea ele sînt specifice fiecărui tip de calculator. Pentru cele bazate pe microprocesoarele 8086 și 8088 (IBM-PC compatibile), firma americană Microsoft Corporation a realizat sistemul de operare MS-DOS (Microsoft-Disk Operating Sistem) ajuns la versiunea 4. 0. Acesta este păstrat pe un suport de memorie externă numit disc sistem. Utilizatorul își poate copia sistemul pe orice disc flexibil sau pe un disc rigid (dacă calculatorul îl con-

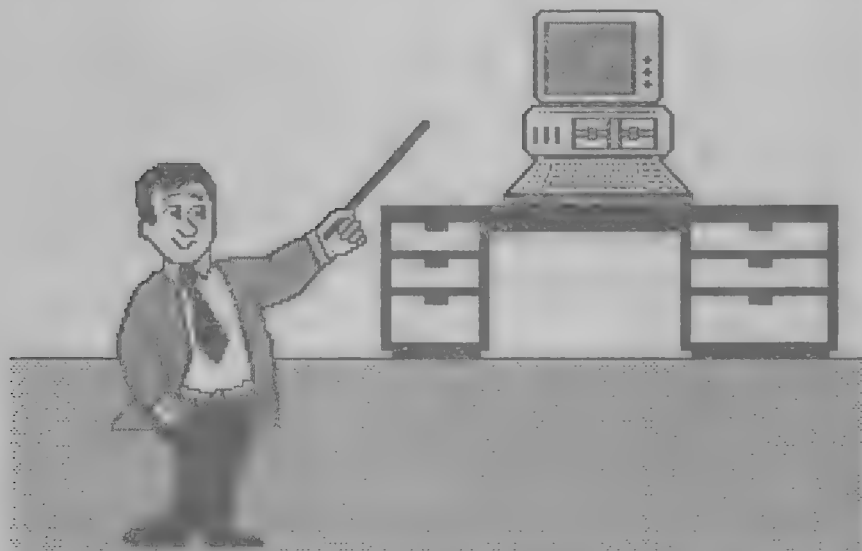
ține în configurație). (El este specific familiei de procesoare pe 16 biți 8086/8088 și nu poate fi utilizat pentru alte tipuri ca Z 8000 de exemplu). MS-DOS asigură comunicarea cu calculatorul, unitățile de discuri, imprimanta, mouse-ul și alte periferice, gestionînd aceste resurse.

El dă posibilitatea utilizatorului: să creeze și să prelucereze fișiere; să înlănțuiască și să execute programe; să acceseze echipamente periferice atașate calculatorului.

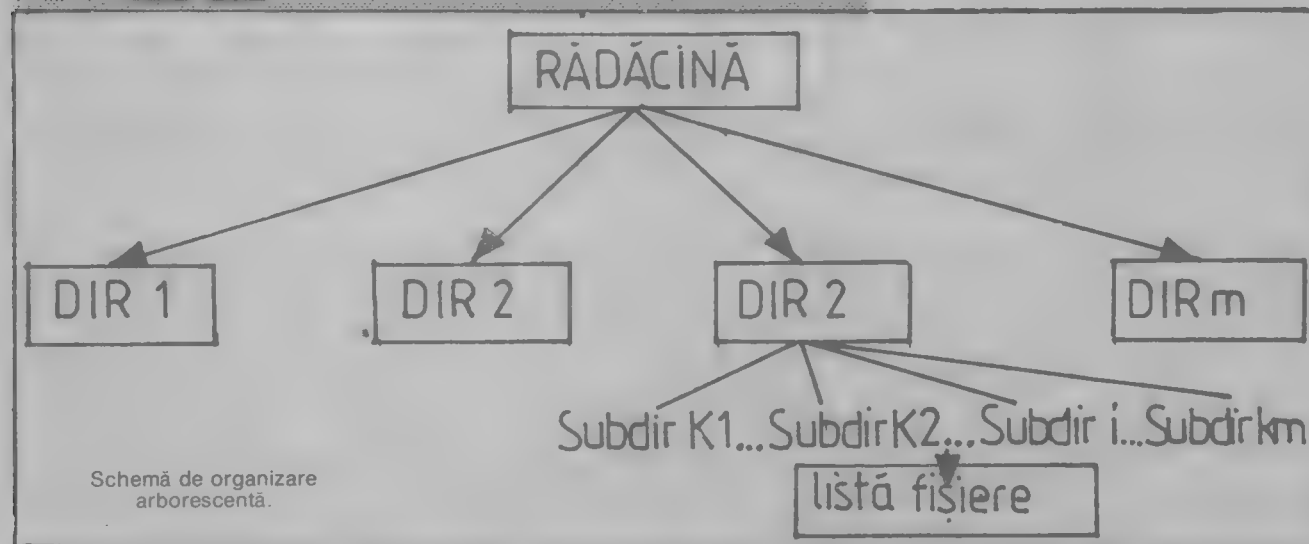
Cea mai mare parte a informațiilor (date și instrucțiuni) memorate pe suporturi externe sînt organizate în fișiere. Un fișier este o colecție de infor-

mații de același tip. Discul sistem conține următoarele fișiere MS-DOS (care conțin comenzi ale sistemului de operare sau informații necesare acestuia):

ANSI.SYS	GRAPHIC.COM
ASSIGN.COM	JOIN.EXE
ATTRIB.EXE	LABEL.EXE
BACKUP.EXE	LINK.EXE
BASIC.COM	MODE.COM
BASICA.COM	MORE.COM
BASICA.EXE	PARK.COM
CHKDSK.COM	PRINT.COM
COMMAND.COM	RECOVER.COM
COMP.COM	RESTORE.COM
DEBUG.COM	SHARE.EXE
DISKCOMP.COM	SORT.EXE
DISKCOPY.COM	SUBST.EXE
EDLIN.COM	SYS.COM
EXE2BIN.EXE	TREE.COM
FDISK.COM	VDISK.SYS
FIND.EXE	MSDOS.SYS
FORMAT.COM	IO.SYS



MS-DOS permite organizarea fișierelor pe discuri în directoare (moduri de împărțire a fișierelor în grupuri, în funcție de opțiunea utilizatorului). Un director poate conține un număr de fișiere dar și alte directoare numite subdirectoare. Astfel se poate obține o structură ierarhică de directoare de tip arborescent care se poate mări, prin crearea unor noi directoare (respectiv subdirectoare).



Discul pe care se lucrează se numește curent, iar directorul în care se lucrează, director curent. Dacă se face o referire la un fișier memorat pe un disc sau într-un alt director, decât cel curent, acestea trebuie specificate prin:

- identificatorul (litera) de disc urmat de simbolul „:” și
- drumul, din arborele director al discului, prin care se ajunge la fișier.

## Introducere în MS-DOS

Pentru inițializare se introduce discul sistem în unitatea de disc logic A (în manualul calculatorului se specifică care este aceasta) și se deschide monitorul și calculatorul. Discul va fi citit și anumite fișiere (nucleul sistemului) care conțin informații necesare permanent sînt introduse în memoria calculatorului. Pe ecran apare mesajul:

**Current date is True 1-01-1980**  
**Enter name date (mm - dd - yy):**  
 MS-DOS așteaptă de la utilizator

## Comenzi MS-DOS

Comenzile sînt căi de comunicare cu calculatorul. O comandă MS-DOS se introduce la terminal pe o linie numită linie de comandă. Prin apăsarea tastei ENTER comanda este automat trimisă la procesorul de comandă (COMMAND.COM) pentru execuție. În același timp o copie este trimisă către o memorie numită linie tampon de unde comanda poate fi rechemată și modificată prin utilizarea unor taste speciale de editare MS-DOS.

Comenzile MS-DOS realizează ur-

Exemplu: A:/DIR 2/SUBDIR 3/ nume fișier extensie, unde nume fișier este format din cel mult 8 caractere (litere, cifre, „-“), iar extensia din cel mult 3 caractere, alese de utilizator.

Directoarele conțin și informații referitoare la dimensiunea fișierelor, poziția lor pe disc și datele cînd au fost create sau actualizate. O zonă adițională a sistemului, creată pe fiecare disc, este tabela de alocare fi-

șiere (File Allocation Table). În ea sînt localizate fișierele pe disc; astfel pot fi alocate spații libere pentru a crea noi fișiere. Aceste două zone sistem, directoarele și tabela de alocare, ajută ca MS-DOS să recunoască și să organizeze fișiere pe discuri. La inițializarea unui disc, MS-DOS crează tabela de alocare și un director gol numit directorul rădăcină.

Următorul mesaj este:  
**Microsoft (R) MS-DOS (R) Version 3.30**

**(c) Copyright Microsoft Corp 1981-1987**

**A>-**  
 În acest moment discul curent este A și se așteaptă o comandă de la utilizator. Dacă se dorește schimbarea unității curente de disc se tastează identificatorul său urmat de „:”. Exemplu: A>B: <ENTER> și pe ecran apare B>- și deci discul B devine disc curent.

data scrisă după formatul din paranteză. Exemplu: Data de 7 Martie 1990 este descrisă prin 07-03-90, după care se apasă tasta ENTER (sau RETURN la unele calculatoare). Precizăm că notația <nume> specifică acțiunea apăsării tastei nume.

Pe ecran va apare un nou mesaj:  
**Current time is 0:00:45:10**  
**Enter new time:**  
 și se așteaptă introducerea timpului. Exemplu: Ora 13 și 30 minute este descrisă prin 13:30 <ENTER>

mătoarele funcții:

- Comparatie, copiere, afisare, ștergere și redenumire fișiere;
- Copiere și formatare (Inițializare) discuri;
- Executare programe de sistem;
- Analiză și listare directoare;
- Introducere dată, timp și comentarii;
- Inițializare opțiuni de imprimantă sau ecran;
- Copiere fișiere sistem MS-DOS pe un alt disc;

• Cererea către MS-DOS de a aștepta o perioadă anume de timp.

Comenzile sistemului sînt de două tipuri: interne și externe. Comenzile interne sînt cele mai simple și mai des utilizate, fiind părți ale procesorului de comandă care este încărcat în memorie. De aceea, cînd sînt tastate, ele se execută imediat. În continuare prezentăm o listă a acestora (în paranteză sînt denumiri echivalente):

BREAK  
 CHDIR(CD)  
 CLS  
 COPY  
 CTTY  
 DATE

DEL (ERASE)  
 DIR  
 ECHO  
 EXIT  
 FOR  
 GOTO  
 IF

MKDIR (MD)  
 PATH  
 PAUSE  
 PROMPT  
 REM  
 REN(RENAM)  
 RMDIR(RD)

SET  
 SHIFT  
 TIME  
 TYPE  
 VER  
 VERIFY  
 VOL

Comenzile externe se află pe disc ca fișiere program. După tastarea unei comenzi de către utilizator, calculatorul trebuie să o citească. Dacă nu o găsește pe disc, MS-DOS nu poate

executa comanda și semnalează printr-un mesaj de eroare. Orice fișier de extensii COM, EXE sau BAT poate fi considerat o comandă externă.

Observație: introducerea unei co-

menzi externe nu trebuie să includă și extensia de fișier. Lista comenzilor externe regăsite pe discul sistem este:

ASSIGN  
 ATTRIB  
 BACKUP  
 CHKDSK

COMP  
 DISKCOMP  
 DISKCOPY  
 EXE2BIN

FIND  
 FORMAT  
 GRAPHICS  
 JOIN

LABEL  
 MODE  
 MORE  
 PARK

PRINT  
 RECOVER  
 RESTORE  
 SHARE

SORT  
 SUBST  
 SYS  
 TREE

# UTILIZAREA SISTEMULUI TURBO PASCAL VERSIUNILE 5.0 și 5.5

## 1. Fișiere necesare utilizării versiunilor 5.0 sau 5.5.

În versiunile 5.0 și 5.5, utilizatorul are la dispoziție două sisteme TURBO Pascal și anume:

- un mediu integrat de dezvoltare (TURBO.EXE), care prin intermediul unor meniuri și ferestre permite editarea, compilarea, execuția și depanarea programului;

- un compilator independent (TPC.EXE), ocupând numai 225 kocteți de memorie, numit *compilator în linie de comandă*.

Pentru utilizarea deplină a tuturor facilităților acestor versiuni mai sînt necesare următoarele fișiere:

- biblioteca de unități standard TURBO.TPL, care cuprinde unitățile System, Crt, Dos, Overlay și Printer.
- unitatea grafică GRAPH.TPU;

- unitățile de compatibilizare cu versiunea 3.0: GRAPH3.TPU și TURBO3.TPU;

- textul help inclus în mediul de dezvoltare (TURBO.HLP);
- driverele dispozitivelor grafice .BGI;

- bibliotecarul TPUMOVER.EXE, permițînd adăugarea și scoaterea de unități din biblioteca de unități standard TURBO.TPL;

- utilitarul TCONFIG.EXE pentru conversia între fișierele de configurare TURBO.TP și TPC.CFG utilizate de către cele două compilatoare;

- utilitarul TINST.EXE pentru instalarea mediului integrat de dezvoltare;

- utilitarul BINOBJ.EXE pentru conversia fișierelor din format binar în format .OBJ;

## 2. Instalarea mediului integrat de dezvoltare TURBO Pascal.

Instalarea TURBO Pascal-ului pe dischete sau disc dur Winchester presupune despachetarea fișierelor corespunzătoare (din formatul compact .ARC) și gruparea lor în subdirectoare, operație care se face uzual cu utilitarul INSTALL; în lipsa acestuia se copiază fișierele, care au fost în prealabil decompactate cu UNPACK. Pentru funcționarea corectă a sistemului TURBO Pascal se recomandă ca în fișierul de configurare CONFIG.SYS să se facă asigurările: FILES=20 și BUFFERS=20.

Mediul integrat de dezvoltare poate fi

folosit fără a-i face nici o modificare copiind fișierul TURBO.EXE de pe discul de distribuție.

Adaptarea mediului integrat de dezvoltare TURBO Pascal se face cu utilitarul TINST, care permite următoarele:

- stabilirea unor căi de acces la subdirectoarele în care sînt plasate fișierele incluse, fișierele de configurare, unitățile și fișierele executabile;

- personalizarea editorului de text TURBO Pascal;

- modificarea opțiunilor implicite ale compilatorului, editorului de legături și debuggerului;

- modificarea culorilor;

- schimbarea dimensiunilor ferestrelor.

Pot fi generate mai multe copii de sisteme de dezvoltare cu diferiți parametri. Programul de instalare se lansează prin:

**TINST [cale\] [/B]**

Copia adaptată a fișierului TURBO.EXE, avînd numele specificat va fi plasată în subdirectorul precizat prin calea de acces. Parametrul /B forțează afișarea monocromă. TINST este comandat prin meniuri. Ieșirea din TINST se face prin acționarea tastei «ESC» din meniul principal de instalare; acționarea aceluiași taste dintr-un submeniu face revenirea la meniul ierarhic superior.

Selectarea unei comenzi dintr-un meniu se face poziționînd, prin intermediul tastelor cu săgeți o linie cursor pe comandă dorită și acționînd apoi «ENTER» sau tastînd litera corespunzătoare comenzii alese.

Meniul principal de instalare dispune de următoarele comenzi (submeniuri):

- *Compile* permite specificarea numelui implicit al Fișierului Primar compilat și a destinației compilării (în Memoria sau pe Disc);

- *Options* permite setarea unor valori implicite pentru opțiunile de compilare (verificarea domeniului, verificarea depășirii stivei, verificarea erorilor de intrare/ieșire, alinierea datelor la limita de cuvînt, evaluarea optimizată a expresiilor booleene, procesarea numerică prin software, emularea coprocesorului, generarea de informații pentru debugger, dimensionarea stivei);

- *Linker* pentru setarea unor opțiuni implicite ale Editorului de Legături;

- *Environment* posedă subcomenzi pentru salvarea automată a fișierului de configurare la folosirea comenzilor **Run**, **File OS Shell** sau **File/Quit**, salvarea fișierului editat la folosirea acelo-

rași comenzi, păstrarea penultimei versiuni a fișierului editat — fișierul .BAK, suprimarea efectului de lupă pentru fereastra activă, care nu va mai ocupa întreg ecranul, setarea unor opțiuni implicite pentru editorul de text;

- *Directory* permite specificarea căilor la subdirectoarele în care se află fișierul de configurare, fișierul help și fișierele unități.

- *Debug* fixează opțiunile implicite ale debuggerului încorporat în mediul integrat de dezvoltare;

- *Editor* permite redefinirea comenzilor editorului;

- *Mode for Display* asigură folosirea unui din modurile: Color, alb-negru, monocrom, LCD;

- *Set Colours* permite modificarea culorilor;

- *Resize Windows* modifică dimensiunile ferestrelor Edit și Output Watc;

- *Quit/Save* permite ca modificările operate să fie instalate permanent în TURBO Pascal;

## 3. Utilizarea mediului integrat de dezvoltare TURBO Pascal

Lansarea sistemului TURBO Pascal se poate face folosind dischetă sau disc dur Winchester cu comanda:

**TURBO**

sau forma mai generală:

**TURBO [fișier—sursă] [/C fișier configurare] [/B] [/D] [/M] [/P]**

În acest din urmă caz:

- fișierul sursă indicat va fi încărcat în editor;

- parametrii de configurare vor fi extrași din fișierul de configurare precizat (implicit aceștia sînt preluați din TURBO.TP);

- /B declanșează opțiunea Build — de recompilare a surselor, fișierelor incluse și unităților folosite;

- /M declanșează opțiunea Make — de recompilare selectivă după dată a sursei, fișierelor incluse și unităților folosite;

Aceste două opțiuni conduc la ieșirea din mediul integrat de dezvoltare.

- /D permite funcționarea cu două adaptoare grafice conectate la două monitoare: un ecran primar pentru ieșirile programului și un ecran secundar pentru mediul integrat TURBO. Comutarea între ele se face prin «Alt»—«F5»;

- /P memorează paleta de culori curentă la comutarea ecranului;

TURBO Pascal este comandat de meniuri și parametri necesari la compilare, fixați prin intermediul acestor meniuri.

Valorile parametrilor astfel modificate vor înlocui în fișierul de configurare valorile fixate la instalare cu TINST.

Meniul principal are forma:

(tabelul 1)

O comandă din meniul principal se alege folosind inițiala numelui ei sau linia cursor, mutată cu tastele săgeți și după care se acționează <Enter>.

Toate comenzile din meniul principal (cu excepția comenzii Edit) deschid alte submeniuuri cu subcomenzi specifice, care se selectează cu o linie cursor poziționată cu ↑ sau ↓ și acționare <Enter>.

Dintr-un submeniu se poate ieși:

- în submeniul ierarhic superior cu <Esc>;
- în meniul principal cu <Alt> <litera>

(litera fiind F, E, R, C, O, D, B)

— în DOS cu <Alt> <X>;

Pe ecran apar trei ferestre: Edit, Watch și Output dintre care una va fi activă.

Fereastra Edit devine activă:

- la selectare E în linia cursor și la acționare <Alt> <E> din alt meniu.

Acționare <Esc> din meniul principal conduce la precedenta fereastră activă; <F6> servește pentru comutarea între fereastra Edit și Output/Watch, în timp ce comutarea între acestea se face cu <Alt>

<F6>. (fig. 1)

#### Meniul FILE — gestione fișiere

Servește pentru încărcarea și salvarea programelor, înregistrarea programelor sub alt nume, declanșare comenzi DOS (fig. 2).

**Load.** (F3) Încarcă un program în editor. În fereastra care se deschide se indică numele programului. Se pot folosi și caractere generice (\* și ?) afișându-se lista fișierelor corespunzătoare. Pentru selectarea unui subdirector se introduce <Shift> și prima literă a numelui acestuia.

**Pick** (Alt-F3). Încarcă în editor fișierul specificat din lista celor mai recent folosite 8 fișiere (Lista Pick).

**New.** Șterge memoria de lucru, pregătind editarea unui nou program.

**Save.** (F2) Salvează pe disc programul în curs de editare sub numele dat fișierului în momentul încărcării.

**Write to.** Salvează programul în curs de editare sub un nou nume.

**Director.** Afișează directorul fișierelor. Se pot folosi caractere generice pentru a selecta fișiere. Pot fi listate și subdirectoare.

**Change dir.** Afișează directorul curent și permite modificarea unității și subdirectorului selectat.

**OS Shell.** Apelează interpretorul de comenzi COMMAND.COM, permițând executarea unei comenzi Dos. Revenirea în TURBO Pascal se face cu EXIT.

**Quit.** (Alt-X). Termină sesiunea TURBO Pascal.

#### Meniul RUN — execuție program.

Comandă execuția unui program de sine stătător sau împreună cu debuggerul, după care se revine în meniul principal (fig. 3).

**Run** (Ctrl-F9). Apelează comanda MAKE înaintea execuției programului.

**Program reset** (Ctrl-F2). Întrerupe funcționarea debuggerului și eliberează memoria ocupată de acesta.

File	Edit	Run	Compile	Options	Debug	Break/Watch
Line 1	Col 1	Insert	Indent	Unindent	A:\NONAME.PAS	
— Watch —						
F1-Help	F5-Zoom	F6-Switch	F7-Trace	F8-Step	F9-Make	F10-Menu

tabelul 1

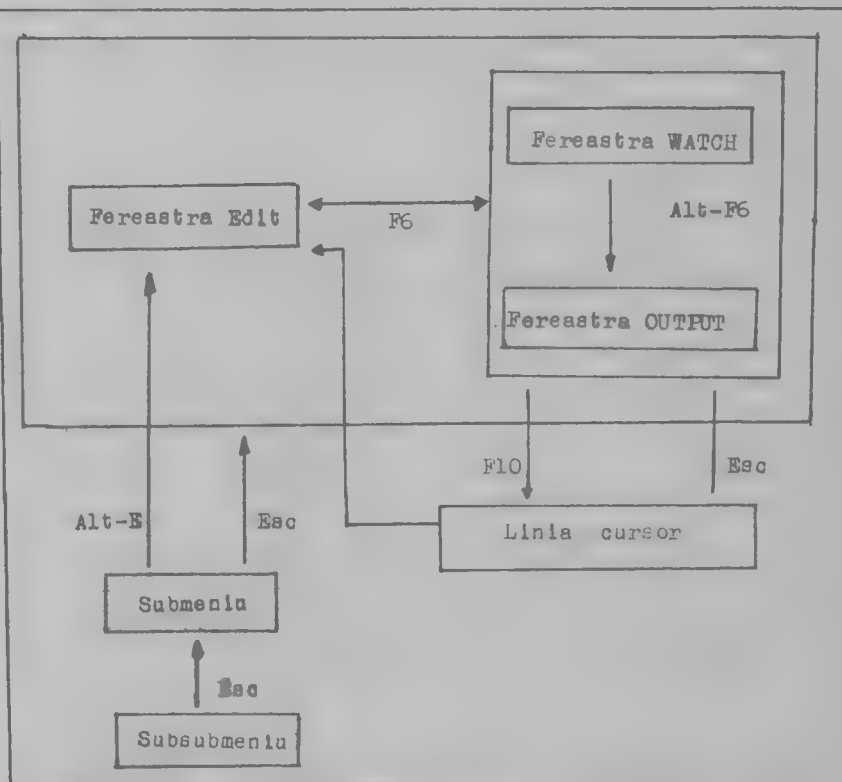


figura 1

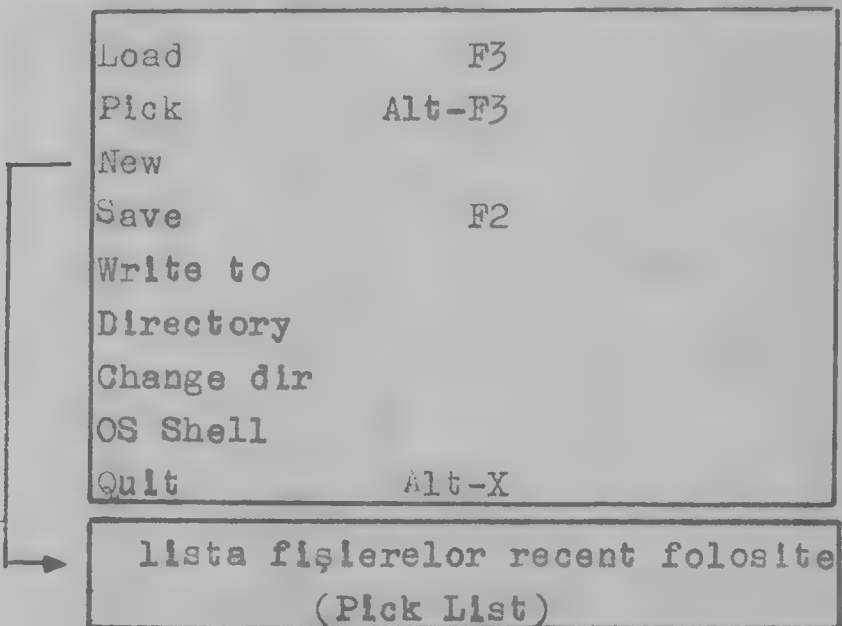


figura 2



Run	Ctrl-F9
Program reset	Ctrl-F2
Go to cursor	F4
Trace into	F7
Step over	F8
User screen	Alt-F5

figura 3

**Go to cursor (F4).** Execută programul până în linia în care se află cursorul.

**Trace into (F7).** Execută programul pas cu pas. Procedurile și funcțiile apelate sint de asemenea parcurse pas cu pas.

**Step over (F8).** Execută programul pas cu pas. Procedurile și funcțiile apelate sint executate „dintr-o dată”.

**User screen (Alt-F5).** Afișează ecranul de execuție (cu rezultatele programului).

**Meniul COMPILE** — compilare program. (fig. 4)

Compile	Alt-F9
Make	F9
Build	
Destination	Memory
Find error	
Primary file	
Get info	

figura 4

**Compile (Alt-F9).** Compilează programul în curs de editare; nu apelează utilitățile MAKE sau BUILD.

**Make (F9).** Apelează utilitarul MAKE: toate fișierele OBJ și INCLUDE sint testate privind data ultimei actualizări — cele care au fost modificate între timp vor fi recompilate.

**Build.** Apelează utilitarul BUILD: acesta recompilază toate fișierele UNIT, OBJ și INCLUDE.

**Destination.** Precizează unde se depune rezultatul compilării: în Memorie sau pe Disc într-un fișier .EXE.

**Find error.** La producerea unei erori la execuție se indică codul erorii și adresa acesteia. Această informație servește pentru localizarea erorii în codul sursă.

**Primary file.** Specifică fișierul .PAS care va fi compilat, independent de fișierul care a fost editat.

**Get info.** Dă informații asupra programului în curs: mărimea fișierilor sursă, numărul de linii, dimensiunea stivei, heapului, etc.

**Meniul OPTIUNI** — fixarea parametrilor de compilare (fig. 5)

**Compiler.** Această opțiune permite parametrizarea compilării. Parametrii se dau sub forma unor opțiuni de activare/inhibare (ON/OFF) sau a unor moduri. Toate aceste opțiuni corespund unor directive de compilare care pot fi fixate global prin meniuri sau local în fiecare program. (fig. 6)

Compiler  
Linker  
Environment  
Directories  
Parameters  
Save options  
Retrieve options

figura 5

Opțiunea din meniu	Valoarea implicită	Directiva comp.
Range checking	Off	\$R
Stack checking	On	\$S
I/O checking	On	\$I
Force far calls	Off	\$F
Align data	Word	\$A
Overlays allowed	Off	\$O
Var-string checking	Strict	\$V
Boolean evaluation	Short Circuit	\$B
Numeric processing	Software	\$N
Emulation	On	\$E
Debug information	On	\$D
Local symbols	On	\$L
Conditional defines		
Memory sizes		\$M

figura 6

**Range checking (off).** Activează sau inhibă verificarea domeniilor: încadrarea indicilor tablourilor în limite, a tipurilor ordinale în subdomenii;

**Stack checking (on).** Când opțiunea este activă, compilatorul generează cod de verificare a încadrării variabilelor locale în spațiul rezervat în stivă;

**I/O checking. (on).** Când opțiunea este activă, o eroare într-o operație de intrare/ieșire produce o eroare la execuție.

**Force far calls (off).** Când opțiunea este inhibată sint permise numai apeluri scurte (în același segment).

**Align data (word).** Variabilele și constantele cu tip sint aliniate la limite de cuvint.

**Overlay allowed (off).** Inhibă generarea de cod pentru segmentare; unitățile de program nu pot fi suprapuse.

**Var string checking (strict).** Verifică concordanța tipurilor string parametri formali și actuali ai procedurilor și funcțiilor.

**Boolean evaluation (Short circuit).** Generează cod pentru evaluarea optimizată a expresiilor booleene.

**Numeric processing (Software).** Permite numai tipul real pe 6 octeți. Tipurile reale single, double, extended și comp sint permise numai în prezența coprocesorului 8087.

**Emulation (On).** Activează/inhibă le-

gătura cu biblioteca run-time care emulează coprocesorul numeric 8087, dacă acesta lipsește.

**Debug information (On).** activează generarea de informație pentru depanare:

**Local symbols (On).** Activează/inhibă generarea numelor și tipurilor variabilelor și constantelor locale dintr-un modul.

**Conditional defines.** Definește simbolurile care pot fi referite în directivele de compilare condiționată.

**Memory sizes.** Permite specificarea explicită a necesarului de memorie pentru program și anume dimensiunea stivei (până la 65535 octeți, implicit se iau 16384 octeți) și limitele inferioară și superioară ale memoriei dinamice — heapul (maxim 655360 octeți).

**Linker.** Această opțiune comandă

funcționarea editorului de legături.

**Map file (Off).** Specifică dacă se dorește generarea unui fișier MAP care conține informații asupra segmentelor, simbolilor publici și adreselor lor și a punctului de intrare.

**Link buffer.** Indică editorului de legături să folosească memoria sau discul pentru păstrarea temporară a informațiilor necesare.

**Environment.** Comandă prin parametri de tip ON/OFF funcționarea mediului integrat de dezvoltare.

**Edit auto save ON** — programul în curs de editare este salvat în mod automat înaintea fiecărei comenzi Run.

**OFF** — programul editat nu este salvat automat, astfel că în cazul unei erori la execuție se pierde.

**Configuration auto save ON** — în fișierul de configurare se înregistrează în mod automat modificările aduse parametrilor.

**Backup files On** sistemul păstrează o copie de siguranță a ultimei versiuni sursă editate (fișierul .BAK).

**Tab size.** Indică numărul de coloane care separă două tabstopuri succesive. Sint permise valori între 2 și 16.

**Zoom windows.** Activarea acestei opțiuni (ON) permite mărirea ferestrei de editare EDIT și a ferestrei rezultatelor OUTPUT la dimensiunea întregului ecran.

**Screen size.** Permite alegerea numărului de linii pentru ecranele EGA și VGA.

**Directories.** Indică subdirectoarele în care se face căutarea și salvarea fișierelor utilizate.

**Turbo directory.** Precizează numele subdirectorului care conține fișierele mediului integrat de dezvoltare.

**EXE & TPU director.** Precizează numele subdirectorului care conține fișierele EXE, TPU și MAP.

**Include directories.** precizează numele subdirectorului care conține fișierele INCLUDE.

**Unit directories** numele subdirectorului care conține fișierele unități.

**Object directories** — numele subdirectorului care conține fișierele OBJ incorporate prin directiva de compilare (SL fișier).

**Pick file name** — permite precizarea unei liste cu numele celor mai recent activate fișiere (lista Pick).

**Save options/Retrieve options.** Permite refacerea sau salvarea unei parametrizări date.

### Meniul DEBUG

Comandă funcționarea depanatorului integrat și gestionează accesul la variabile (fig. 6).

Evaluate	Ctrl-F7
Call stack	Ctrl-F8
Find procedure	
Integrated debugging	On
Stand-alone debugging	Off
Display swapping	Smart
Refresh display	

figura 7

**Evaluate** — afișează un cadru de dialog cu trei zone pentru afișarea și manipularea variabilelor. **Evaluate** permite indicarea unei variabile sau expresii a cărei valoare va fi indicată în zona rezultatelor (Output).

**Result.** Afișează valoarea expresiei sau conținutul variabilei selectate prin **Evaluate**.

**New value.** Permite indicarea unei noi valori în zona **Evaluate**.

**Call stack.** Permite încărcarea în editor a unei proceduri sau funcții apelate de către program, selectată dintr-o listă de rutine apelate printr-o linie cursor.

**Find procedure.** Crează un cadru de dialog, permițând precizarea unui nume de procedură sau funcție care va fi încărcată în editor.

**Integrated debugging.** Indică generarea de informație de depanare necesară folosirii depanatorului integrat.

**Standalone debugging.** Informațiile de depanare generate de compilator vor fi înglobate în fișierul EXE creat pe discetă.

Are sens în cazul folosirii unui depanator extern independent.

**Display swapping.** Precizează modul în care depanatorul afișează ecranul de execuție.

**Smart** — ecranul de execuție apare numai dacă instrucțiunea modifică afișarea.

**Always** — ecranul de execuție apare după fiecare instrucțiune.

**None** — ecranul de execuție este afișat numai la acționare <Alt> <FS>.

**Refresh display** — reface complet afișarea mediului de dezvoltare.

### Meniul BREAK/WATCH

Când se lucrează cu depanator integrat, meniul Break/Watch permite adăugarea sau ștergerea de variabile martor din fereastra Watch și plasarea și ștergerea de puncte de interpretare. (fig. 7)

Add watch	Ctrl-F7
Delete watch	
Edit watch	
Remove all watches	
Toggle breakpoint	Ctrl-F8
Clear all breakpoints	
View next breakpoint	

figura 8

**Add watch** — permite adăugarea unei noi variabile martor (Watch) în fereastra Watch; când aceasta este activă se pot adăuga variabile martor folosind testele de editare <Ctrl> <N> sau <Ins>.

**Delete watch** — permite ștergerea unei variabile sau expresii martor din fereastra Watch; când aceasta este activă, același rezultat se obține cu tastele <Ctrl> <Y> sau <Del>.

**Edit watch** — permite editarea numelui unei variabile sau a unei expresii martor aflată în fereastra Watch.

**Remove all watches** — șterge toate variabilele și expresiile martori din fereastra Watch.

**Toggle breakpoint** — plasează sau șterge un punct de întrerupere în linia în care se afla cursorul.

**Clear all breakpoints** șterge toate punctele de întrerupere.

**View next breakpoints** — mută cursorul pe următorul punct de întrerupere.

### 4. Facilități ale limbajului TURBO Pascal versiunile 5.0 și 5.5.

Față de versiunile anterioare sistemul TURBO Pascal versiunea 5.0 prezintă următoarele avantaje:

- 1) depanarea la nivelul codului

sursă, incluzând: execuția pas cu pas, puncte de întrerupere, examinarea variabilelor, structurilor de date și expresiilor;

2) emularea virgulei mobile în lipsa coprocesorului matematic 8087 sau 80287;

3) segmentarea bazală pe unități;

4) compatibilitate cu TURBO Debugger;

5) posibilitatea folosirii extensiei de memorie (de către editorul de texte și de către unitatea Overlay);

6) viteza de compilare de 3—5 ori mai ridicată față de versiunea 3.0;

7) generarea de cod îmbunătățită producând o execuție mai rapidă;

8) editor de legături încorporat care înlătură porțiunile nefolosite de cod și date producând programe mai mici;

9) fișiere EXE producând programe care depășesc 64 KOcteți de memorie;

10) compilare separată folosind unitățile de memorie;

11) unități standard: System, Dos, Crt, Overlay, Printer și Graph;

12) interfața mai puternică cu limba-jul de asamblare;

13) posibilitatea de includere a fișierelor pe 8 niveluri de adâncime;

14) tipuri de date noi: shortint, longint, word, single, double, extended, comp;

5) proceduri și funcții standard noi;

16) evaluarea optimizată a expresiilor booleene;

17) directive de compilare condiționate;

18) compatibilitate ridicată cu versiunile 3.0 și 4.0; utilitare pentru conversia programelor între versiuni;

19) două versiuni de compilator: mediu integrat de dezvoltare (editor, compilator, editor de legături, depanator, biblioteca runtime) și compilator cu linie de comandă;

20) recompilarea tuturor unităților (opțiunea Build) sau în mod selectiv după dată (opțiunea Make) la efectuarea unei actualizări într-una din unități sau în program;

### 5. Diferențe între versiunile 5.0—5.5 și 3.0

VERSIUNEA 3.0	VERSIUNILE 5.0 - 5.5
Un identificador din program poate avea același nume cu numele programului	numele programului este unic; orice identificador din program poate fi referit prin nume program. identificador
toate obiectele programului se compilează în același timp extrăgându-se dintr-un fișier sursa și din fișiere incluse	obiectele se grupează în unități care se compilează separat
un program executabil (fișier .GOM) poate avea maxim 64 KO cod	programul executabil (.EXE) este format din unități, fiecare putând avea pînă la 64 KO cod
pentru a depăși restricțiile de memorie se folosește segmentarea și înlănțuirea	facilitățile de segmentare sînt mai sofisticate și mai transparente pentru utilizator. înlănțuirea programelor se face cu procedura Exec din unitatea Dos
opțiunea de includere (\$I nume poate fi plasată oriunde și poate fi formată numai din instrucțiuni	fișierul inclus nu poate conține numai instrucțiuni și proceduri și funcții complete; el nu poate fi plasat între begin și end

fișierele incluse nu pot include alte fișiere	sînt permise incluziuni de fișiere pînă la 9 niveluri de adîncime
intrările de la consola sînt tratate nestandard comparativ cu intrările disc	intrările consola sînt compatibile cu intrările din fișierele disc
	tipuri noi: shortint, longint, word, single, double, extended, comp
constanțele predefinite sînt cunoscute în tot programul	constanțele predefinite în unități sînt accesibile din program numai dacă se folosește clauza uses
IOResult întoarce coduri de eroare specifice versiunii 3.0	IOResult întoarce coduri de eroare specifice MSDOS.
tipuri string numai cu indicație de lungime	string = string (255)
într-o procedură sau funcție variabilă de control a ciclului for poate fi locală sau globală	într-o procedură sau funcție variabilă de control a ciclului for trebuie să fie locală
se pot declara etichete fără a le folosi	etichetele declarate și nefolosite sînt erori sintactice
lungimea variabilei tampon pentru fișiere text poate fi stabilită la declararea variabilei fișier	lungimea variabilei tampon pentru fișiere text poate fi stabilită numai cu SetTextBuffer
constanțele cu tip sînt rezidente în segmentul de cod	constanțele cu tip sînt rezidente în segmentul de date
conversie numai între tipurile ordinale	conversie între toate tipurile simple cu restricția ca sursa și destinația să aibă aceeași lungime
evaluarea completă a expresiilor booleene	evaluarea optimizată a expresiilor booleene
proceduri de tratare a întreprinderilor numai în limbaj de asamblare	proceduri de tratare a întreprinderilor în Pascal folosind directiva interrupt
rutinele externe în limbaj de asamblare trebuie să fie în format .BIN cu offseturi față de prima rutină din fișier	rutinele în limbaj de asamblare pot fi în format .OBJ fiind legate folosind opțiunea \$L fișiere-obiect
apelurile de funcții și proceduri sînt apeluri scurte în același segment de cod	se generează după necesitate apeluri scurte sau lungi \$P+3 forțază apeluri lungi între segmente
tabelul 2	instrucțiunea inline nu permite referiri la conținutul de locații nici la nume de proceduri sau de funcții

## IMPORTANT!

Am primit la redacție: S-a constituit — Fundația „East-West Education Development”, sponsorizată de Patrick McGovern - Președintele IDG - și de către IDG. Scopul principal: sprijinirea țărilor din estul Europei privind accesul la tehnologia informației în îmbunătățirea sistemului de învățămînt și de educație prin intermediul calculatoarelor.

Fundația roagă toți producătorii și comercianții de tehnică de calcul să pună la dispoziția acestora echipamentele la care renunță. Avantajele sînt, credem, cunoscute: scutirea de impozite, reclamă etc. Vă rugăm să ne răspundeți pe adresa redacției, noi urmînd a trimite fundației răspunsul dumneavoastră. Vă mulțumim!

## Computerworld

Recent a fost difuzat un raport cu privire la securitatea informațiilor și a datelor vehiculate prin intermediul calculatoarelor, avînd în vedere proliferarea rețelelor. Raportul, intitulat sugestiv „Computers at risk” a concluzionat că utilizatorii și comercianții din domeniu nu sînt încă suficient de bine informați și nu acordă importanța cuvenită sistemului de protecție comercială. În același timp, raportul a pus bazele unui sistem de protecție a informației (GSSP — Generally Accepted System Security Principles) care reprezintă un ghid de protecție pentru utilizarea calculatoarelor în rețele. Acest raport a fost conceput de un comitet pentru studierea sistemelor de securitate, afiliat la Academia Națională de Științe din S.U.A. ca un răspuns la cererea formulată de DARPA (Defense Advanced Research Projects Agency). Din acest comitet fac parte 16 membri implicați în producția sau cercetarea din domeniu de la instituții importante, dintre care: AT and T Bell Laboratories, BBN Communications Corp, Digital Equipment Corp și altele.

Iată cîteva dintre recomandările raportului, în afara celor menționate mai sus:

- determinarea utilizatorilor și distribuitorilor de a utiliza cit mai repede și eficient sistemele deja existente de securitate;

- alcătuirea unei baze de date cu privire la toate informațiile existente despre securitatea datelor și tranzacțiilor și încurajarea specializării în aceste sisteme de securitate și de... etică profesională;

- simplificarea și sistematizarea restricțiilor guvernamentale existente cu privire la comercializarea sistemelor de securitate și codificare a datelor peste ocean.

- înființarea unui fond special pentru cercetările în domeniul securității informațiilor și protecțiilor. (Network World/ 3 decembrie 1990/ 7 ianuarie 1991, Digital News/ 7 ianuarie 1991).

## Computerworld

# NeXT DIN NOU ÎN ACTUALITATE

Traducere și prelucrare de Mihaela GORODCOV după articolul „NEXT ON THE AGENDA” de Bruce F. Webster apărut în numărul din ianuarie 1991 al revistei MACWORLD.

Cu doi ani în urmă, anunțam în paginile revistei „Știință și tehnică” și ale almanahului cu același nume, apariția în „forță” a unui nou tip de calculator cu multe inovații, care se anunța deosebit de promițător: sistemul NEXT. Steve Jobs — unul dintre „copiii teribili” ai informaticii și cofondatorul — alături de Steve Wozniak, al firmei Apple, a creat NeXT-ul ca o mașină îndrăzneată, lansată cu o campanie publicitară corespunzătoare în octombrie 1988. Mănușa fusese aruncată! NeXT-ul, cel controversat, urma să revoluționeze din multe puncte de vedere modul de lucru „personal” al utilizatorului care avea acum la dispozi-

ție o mașină puternică, cu multe facilități și deschideri spre numeroase aplicații.

Firește că la NeXT se regăseau multe dintre posibilitățile calculatoarelor Apple Macintosh, precum și ale altora. Desigur că NeXT-ul avea, de asemenea, limitările sale: lipsa software-ului comercial, prețuri destul de ridicate și lucrul destul de lent. Asupra altor caracteristici ale NeXT-ului din octombrie 1988 nu mai revenim. Ele au fost descrise pe larg în revista și în almanahul ST.

Acum, după doi ani de la lansare, nimeni nu poate spune că NeXT nu a „învățat” ceva din propriile greșeli. Re-

venirea din septembrie 1990 a fost făcută tot „în forță”. Compania a introdus o linie de sisteme care nu numai că sînt mai puternice, dar și mai... ieftine decît mașina inițială. Înalta rezoluție și larga paletă de tonuri de gri a monitoarelor alb/negru ale modelului inițial a fost acum completată cu două sisteme color. Mai mult decît atât, două dintre companiile faimoase de software și-au introdus produsele lor pe NeXT: IMPROV (LOTUS) și WORDPERFECT de la firma cu același nume. Și, pentru a liniști „vocile” care acuză numărul relativ mic de mașini instalate, NeXT a anunțat că are deja comenzi ferme pentru 15 000 de

	Mac IIx System	List Price	Nextcube System	List Price
Basic System	4MB RAM, 160MB hard drive	\$10,969	8MB RAM, 340MB hard drive	\$9000
CPU	40MHz 68030		25MHz 68040	
MMU	built-in to 68030		built-in to 68040	
FPU	40MHz 68882		built-in to 68040	
DSP	none		25MHz 56001	
I/O processors	1 SCSI DMA, 2 serial I/O processors		12 DMA I/O processors	
ROM	512K		NA	
RAM	16MB	\$880	16MB	\$400
Video	Macintosh Display Card 8*24	\$899	1120 x 832 x 4 shades (built-in)	
Monitor	Apple Two-Page Monochrome Monitor (1152 x 870 x 16)	\$2149	MegaPixel Display (1120 x 832 x 4)	\$995
Keyboard	Apple Keyboard	\$129	Next keyboard	
Sound input	none		8-bit, 8kHz sampling	
Sound-input ports	none		microphone jack, built-in microphone	
Sound output	Apple sound chip with built-in speaker		56001 DSP with built-in speaker	
Sound-output ports	stereo minijack		stereo minijack, dual line-outs	
Other ports	serial (2); SCSI; ADB (2)		serial (2), SCSI/2, printer, DSP, Ethernet	
Slots	6 (4 available)		4 (3 available)	
Floppy drive	1.44MB		2.88MB	
Networking	AppleTalk (built-in), EtherTalk	\$699	Ethernet (built-in)	
System software	A/UX 2.0 (on floppies)	\$995	Release 2.0 extended	
Bundled software	System 6.0.6, HyperCard, Edit, Shell, UNIX utilities		WriteNow, Webster's Ninth New Collegiate Dictionary, Librarian, Mail, Edit, Shell, UNIX utilities, Quotations, Shakespeare, demo applications, NextStep development tools	
Total Retail Price for Monochrome System		\$16,720		\$10,395
Color monitor	Radius 19" Display (1152 x 882)	\$4295	Next MegaPixel Color (1120 x 832)	\$2995
Color video board	Radius DirectColor/24	\$3595	Nextdimension (32-bit color with alpha)	\$3995
Graphics acceleration	Radius QuickColor	\$595	Intel i860 RISC processor	
Video integration	RadiusTV	\$2795	NTSC/S-Video/RGB input and output	
Color compression	color-compression board [Note: no slots available]	\$995	C-Cube JPEG color compression chip [note: two slots available]	
Total Retail Price for 24-bit System		\$28,995		\$16,390

MAC IIx VERSUS NEXTCUBE





sisteme.

Dar să vedem, pe scurt, care sînt membrii familiei NeXt. Au fost lansate deja trei sisteme de bază: NEXTSTATION, NEXTSTATION COLOR și NEXTCUBE la care se pot conecta

cîteva echipamente deosebit de performante: MegaPixel Display, imprimanta laser NeXT cu 400 dpi (dot per inch) — la un preț redus —, extinsă pentru grafică video Nextdimension pe 32 biți și monitorul color de 16 inch MegaPixel Color Display.

Toate cele trei sisteme NeXt intru-nesc calitate și performanțe cu adevărat deosebite:

- Microprocesor 68040 (Motorola) de 25 MHz pe post de CPU (Central Processing Unit), unitatea de gestionare a memoriei (MMU — Memory Management Unit) și unitatea de virgulă mobilă (FPU — Floating Point Unit).

- Procesor pentru prelucrarea digitală a semnalului (DSP) 56001 la 25 MHz cu 24 K memorie în RAM Static expandabilă pînă la 576 K.

- 16 conectoare SIMM pentru memoria principală capabile să accepte de la 1MB pînă la 4MB.

- 2 cipuri VLSI specializate pentru „mainframe” cu rolul de a implementa 8 procesoare I/O (9 pentru Next Cube) și alte facilități de sistem.

- Placă video și RAM — video separat — care generează imagini de 1120 pixeli pe 832 linii, care constituie suportul pentru „alpha-channel”.

- Includerea hardware-ului Ethernet specializat cu conectoarele externe atît pentru BNC T-connector, cît și pentru cablul panglică 10 Base-T, acesta din urmă fiind capabil să suporte conexiuni Ethernet de la fire

standard telefonice.

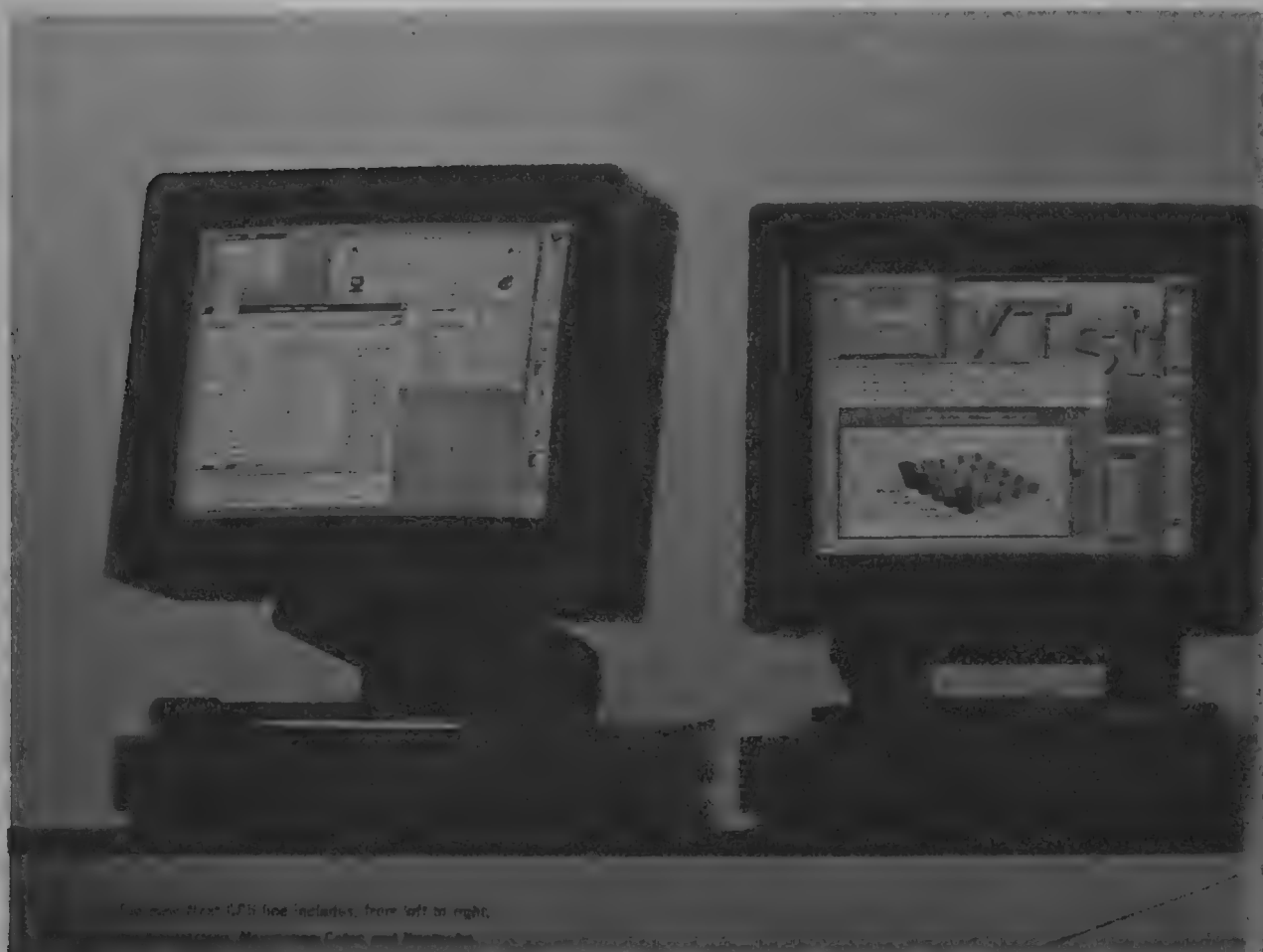
- Un drive de floppy disc de 3 1/2 inch cu capacitate de 2,88 MB care poate, de asemenea, citi și înscrisuri MS-DOS (atît de 720 K cît și de 1,44 MB)

- Porturi duale seriale (DIN-8 RS-422); porturi specializate pentru video și imprimantă; port DSP; interfață-SCSI/2 cu porturi interne și externe.

- Intrare de sunet (inclusă în interiorul mașinii) de 8 biți și 8 KHz, eșantionare și ieșire de sunet — cu canal dublu de 16 biți și 44,1 KHz eșantionare.

- „Release 2.0.” ca sistem software al lui NeXt care include Next Step 2.0. aflat în topul sistemului de operare Mach (UNIX).

Cam acestea ar fi în mare facilitățile și performanțele familiei NeXt, care deschid — după cum se poate observa — un cîmp foarte larg de aplicații multor categorii de utilizatori. În cele ce urmează vă propunem o foarte succintă „fișă tehnică” a fiecărui sistem, cu alte cuvinte cîteva tabele comparative între mașinile NeXt și cele din familia Macintosh Apple. Desigur că nu ne propunem în acest material să epuizăm subiectul NEXT. El este foarte vast și, firește, că vom reveni asupra lui. De asemenea, cu ocazia acestui material facem și o deschidere către familia de calculatoare Apple Macintosh, căreia, în numerele viitoare îl vom consacra niște materiale speciale din documentația „la zi” primită prin rețeaua IDG.



Two new NeXt S.P.0 line computers, from left to right, NeXTcube and NeXTstation.

	Mac IIx System	List Price	Nextstation System	List Price
Basic System	2MB RAM, 40MB hard drive	\$3800	8MB RAM, 105MB hard drive	\$4000
CPU	20MHz 68030		25MHz 68040	
FPU	20MHz 68882	\$200	built-in to 68040	
MMU	built-in to 68030		built-in to 68040	
DSP	none		25MHz 56001	
I/O processors	none		12 DMA I/O processors	
ROM	512K		NA	
System RAM	8MB	\$600	8MB	
Built-in video	640 × 480 × 256 colors/shades 640 × 870 × 16 shades		1120 × 832 × 4 shades (plus 4 levels of alpha)	
Monitor	Apple Portrait Display (640 × 870)	\$1099	MegaPixel Display (1120 × 832)	\$995
Keyboard	Apple ADB Keyboard	\$99	Next keyboard	
Sound input	8-bit, 11kHz or 22kHz sampling		8-bit, 8kHz sampling	
Sound-input ports	microphone jack (microphone included)		microphone jack, built-in microphone	
Sound output	Apple sound chip with built-in speaker		56001 DSP with built-in speaker	
Sound-output ports	stereo minijack		stereo minijack, dual line-outs	
Other ports	serial (2), SCSI, ADB		serial (2); SCSI/2, printer, DSP, Ethernet (thin wire and twisted pair)	
Slots	1		0	
Floppy drive	1.44MB		2.88 MB	
Built-in networking	AppleTalk		Ethernet	
System software	System 6.0.6		Release 2.0	
Bundled software	HyperCard, System utilities		WriteNow, Webster's Ninth New Collegiate Dictionary (small version), Librarian, Mail, Edit, Shell, UNIX utilities	
Total Retail List Price		\$5798*		\$4995

\* With AppleColor High-Resolution RGB Monitor instead (640×480), \$5698.

	Mac IIx System	List Price	Nextstation Color System	List Price
Basic System	4MB RAM, 80MB hard drive	\$6669	12MB RAM, 105MB hard drive	\$4875
CPU	25MHz 68030		25MHz 68040	
MMU	built-in to 68030		built-in to 68040	
FPU	25MHz 68882		built-in to 68040	
DSP	none		25MHz 56001	
I/O processors	none		12 DMA I/O processors	
	512K		NA	
	8MB	\$300	8MB	
Built-in video	640 × 480 × 256 colors		1120 × 832 × 4096 colors (with 4-bit alpha)	
Monitor	AppleColor High Resolution RGB Monitor (640 × 480)	\$999	Next MegaPixel Color (1120 × 832)	\$2995
Keyboard	Apple Keyboard	\$129	Next keyboard	
Sound input	none		8-bit, 8kHz sampling	
Sound-input ports	none		microphone jack, built-in microphone	
Sound output	Apple sound chip with built-in speaker		56001 DSP with Next Sound Box	\$125
Sound-output ports	stereo minijack		stereo minijack, dual line-outs	
Other ports	serial (2), SCSI, ADB (2)		serial (2), SCSI/2, printer, DSP, Ethernet (thin wire and twisted pair)	
Slots	3		0	
Floppy drive	1.44MB		2.88MB	
Built-in networking	AppleTalk		Ethernet	
System software	System 6.0.6		Release 2.0	
Bundled software	HyperCard, System utilities		WriteNow, Webster's Ninth New Collegiate Dictionary (small version), Librarian, Mail, Edit, Shell, UNIX utilities	
Total Retail List Price		\$8097		\$7995

Deși HC-85 este calculatorul românesc care reproduce cel mai fidel posibilitățile lui Sinclair Spectrum, utilizatorii săi și-au putut da seama încă din primele momente că el s-a născut „bolnav”, din punct de vedere hard, BRIGHT nefiind implementat. Această infirmitate ne privează, pe de o parte, la crearea unor programe de utilizarea și a nuanțelor derivate din cele opt culori, iar pe de altă parte, la rularea programelor de firmă, de gama integrală a informației. Un exemplu edificator în acest ultim caz, este imposibilitatea folosirii grilei de BRIGHT în utilitarul „Artstudio”.

Vă propunem remedierea acestei

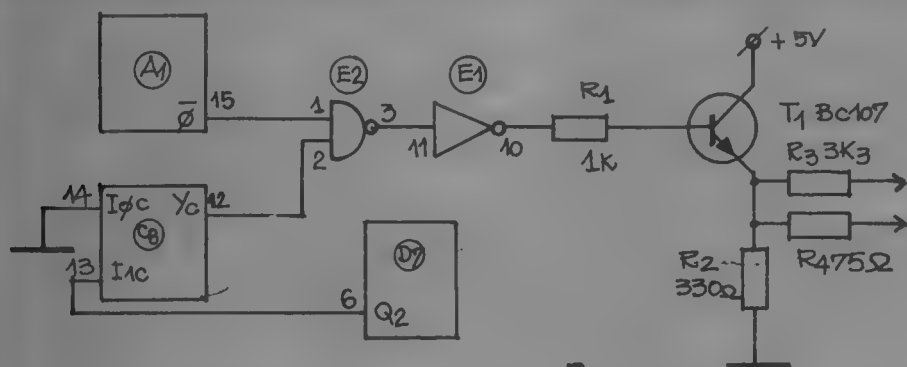
deficiențe prin utilizarea la maximum a circuitelor plăcii de bază. Trebuie remarcat că, suplimentar, se folosesc: un tranzistor NPN, trei rezistoare și circa 0,5 m sîrmă...

Cele de față constituie și o invitație adresată producătorului, ICE, pentru revizuirea plăcii de bază în sensul includerii schemei BRIGHT prezentate. Figura 1 reprezintă schema BRIGHT. Conexiunea între părțile anterior nefolosite ale circuitelor integrate se realizează cu sîrmă izolată pe partea cu piese, exceptînd conexiunea 11 (E1) — 3 (E2) care poate fi făcută pe partea opusă. Lipiturile se fac direct la pinii circuitelor integrate. Este bine să se respecte următoarea ordine:

Pin C.I. Pin C.I.  
1 (E2) — 15 (A1)  
1 (E2) — 12 (C8)  
6 (D7) — 13 (C8)  
14 (C8) — GND (de ex. 15 (C7))

Tranzistorul și cele trei rezistoare vor fi implantate în zona A6 a plăcii, conform schemei din figura 2. Este preferabil să se folosească rezistoare cu peliculă metalică sau chimică de 0,25 W (datorită dimensiunilor reduse). Tranzistorul poate fi BC 107, 108, 109 etc.

A1, C7, C8, D7, E1, E2, R110, R111, R112, R113, T9 sînt denumirile componentelor din schema producătorului.



Spre punctul comun  
R110, R111, R112, R113  
Conector video pin 2

Fig. 1

**Funcționarea schemei.** Semnalul de BRIGHT, bitul 6 al octetului de atribute, este prezent pe pinul 6 al registrului de atribute D7. Prin C8 semnalul este multiplexat cu cel de BORDER și este trimis la poarta E2, tip NAND. Semnalul de BRIGHT este oprit dacă culoarea curentă este 0 (nu există BRIGHT pe negru la Spectrum!) cu ajutorul decodificatorului de culoare din codorul PAL (A1). În continuare, semnalul este inversat prin E1, amplificat de tranzistorul T și aplicat în punctul comun al rezistoarelor R110-R113, unde se formează semnalul Y de luminanță.

Pentru monitoare RGB semnalul trebuie preluat din emitorul tranzistorului T printr-o rezistență de 75 ohmi și aplicat la conectorul video (pinul 2).

Trebuie precizat că la conectarea schemei BRIGHT nivelul de luminanță va scădea puțin, ceea ce poate fi eventual compensat printr-o ușoară modificare a reglajului de nivel din modulatorul TV.

Se recomandă ca modificările prezentate să fie executate de un specialist în domeniu, pentru a nu avea surprize cel puțin neplăcute!

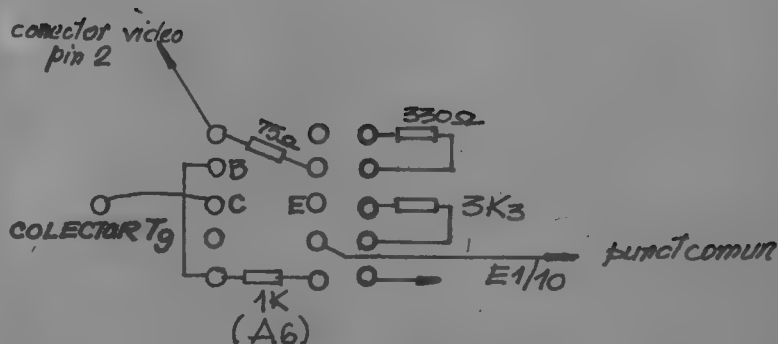


Fig. 2

# UN INSTRUMENT DE LUCRU ÎN REDACȚIA NOASTRĂ: CALCULATORUL

Desigur că, multora dintre dv. acest articol și, mai ales, microcalculatoarele CP/M, li se vor părea depășite și totuși.... Noi îl publicăm, după mai bine de 3 ani, ca un semn de restituire și respect pentru munca unor oameni entuziaști, care au reușit să facă din redacția „Știință și tehnică”, cu toate avatarurile acelor vremuri, prima redacție informatizată din România. Pentru această prioritate și pentru un imens capital sentimental, publicăm acum acest articol, cu toate mulțumirile și recunoștința noastră pentru cei care ne-au ajutat și au crezut în noi.

Mulți utilizatori de microcalculatoare sînt convinși că un editor de texte gen Wordstar satisface orice cerință. Și acestora le sînt adresate explicațiile ce urmează pentru a putea face diferența dintre textele culese în tipografie și cele realizate după programele de tip Wordstar.

Orice text – indiferent de conținutul său – își începe viața de-abia după ce este înregistrat. Operația se poate face pe hîrtie sau pe suport magnetic, dar rămînerea la acest stadiu face din text numai un document primar, chiar și în cazul cînd, în continuare, el este multiplicat prin cele mai diferite mijloace. Majoritatea textelor ajunse la cititor reprezintă însă rezultatul unor îndelungate și repetate prelucrări ce se constituie într-un domeniu cunoscut, dar în intensă și rapidă dezvoltare – activitatea redacțional-editorială.

Pînă în urmă cu cîteva decenii, instrumentele de lucru din redacții erau dintre cele mai simple: creionul (stiloul, pixul), guma, foarfecele și lipiciul. Un instrument ceva mai evoluat era mașina mecanică de scris, în fața căreia, de multe ori, nu se află un redactor, ci doar o dactilografă. Productivitatea muncii cu asemenea mijloace rudimentare este destul de redusă, mai ales dacă se ține seama că un text pornit de la sursă (autorul) suferă cîteva reînregistrări pînă cînd ajunge la destinație (cititorul). Vechiul proces redacțional este reprezentat în schema nr. 1.

După cum se observă, operațiile cele mai redundante sînt înregistrarea textului și corecturile cu un coeficient de repetare de pînă la 2,5-3 ori. În plus, procesul de transmitere a textului de la autor la cititor este mult încetinit în cazul editărilor de cărți și reviste de multiplu schimb de corecturi dintre tipografii și redacții.

## Ce nu știe cititorul

Un text destinat tiparului nu poate fi transmis ca atare tipografiei. El trebuie însoțit de o serie de indicații de tehnoredactare din care enumerăm pe cele mai importante:

- **formatul de bază al rîndurilor** (exprimat în unități de măsură vechi, specifice poligrafiei, de exemplu cicero sau pica, puncte tipografice sau unități relative, cva-drați etc.);

- **familia de litere** (aceasta este de obicei un anume sortiment de semne de tipar elaborat de un grafician specializat în proiectare de alfabete pentru tipar; ele pot fi protejate prin legea dreptului de autor și poartă nume distincte, de exemplu univers,

megaron, bodoni, gill, souvenir etc.);

- **tăietura literei** (este o specificare suplimentară a unei variante a familiei de litere, de exemplu univers drepte, cursive, aldine, subțiri, aldin-cursive etc.);

- **corpul de literă** (este înălțimea literei, exprimată în puncte tipografice);

- **interlinierea** (este măsura distanței dintre rînduri, exprimată în puncte și fracțiuni de puncte tipografice);

- **starea** (mărimea spațiului fix cu care începe un paragraf sau aliniat exprimat în pătrisoari, semipătrisoari, puncte tipografice);

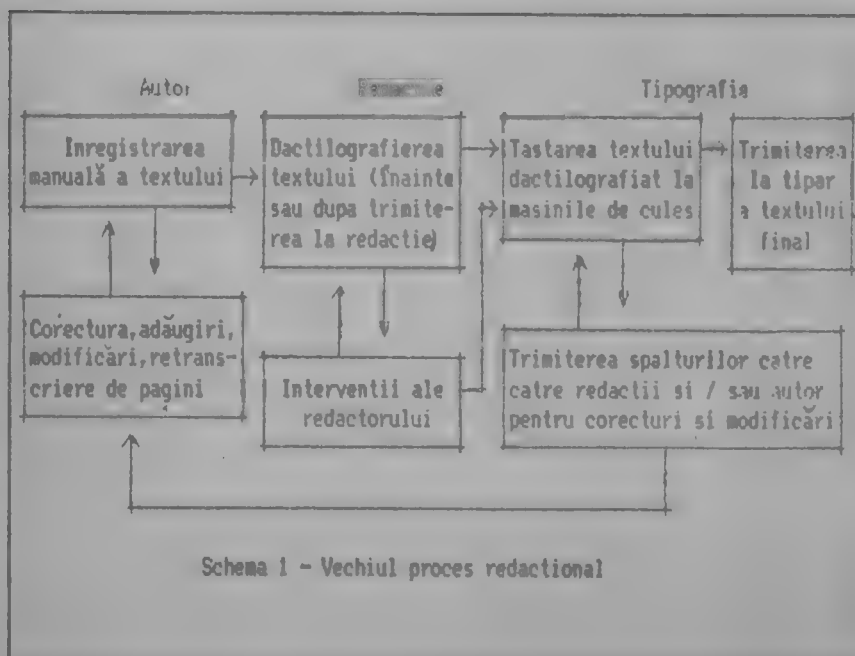
- **specificările** pentru titluri, legende la figuri, note de picior etc.

Toate aceste indicații modelează în fel și chip aspectul textului tipărit și reprezintă componente esențiale ale prezentării poligrafice ce poate fi numită și estetica de carte.

Un text tipărit nu poate fi pus pe picior de egalitate cu un text dactilografat sau listat de imprimantă din considerente de estetică a tiparului (chiar și în cazul celor mai avansate sisteme de birotică, de exemplu Desktop Publishing). Varietatea familiilor și corpurilor de literă este deosebit de bogată în cazul tiparului și extrem de săracă la textele dactilografiate sau listate. Lizibilitatea unui text tipărit este net superioară, mai ales din cauza variației lăților literelor (la tipar aproape fiecare literă poate diferi de alta prin lățime, în timp ce litera sau semnul dactilo sau listat au lățimi egale – de exemplu literele m și i, precum și punctul sau virgula au lățimi egale între ele). În plus, spațiile dintre cuvinte (blancurile) variază la tipar imperceptibil, aproape fără trepte de la rînd la rînd, dar au măriri constante în cadrul aceluiași rînd. Dar... să ne oprim din argumentare, căci criteriile de estetică a tiparului nu pot fi epuizate în cîteva exemple și, de altfel, nu acesta este obiectul articolului de față.

Calculatoarele electronice au adus și în cadrul redacțiilor modificări substanțiale ale stilului de lucru și aceasta, trebuie să recunoaștem, constituie, prin efortul de a se adapta, o dificultate pasageră pentru redactorii zilelor noastre, dar sporește cu mult productivitatea celor implicați în activitatea redacțional-editorială.

Implementarea lor în redacții nu ar fi fost posibilă dacă în domeniul culegerii textelor pentru tipar nu ar fi apărut fotoculegerea – procedeu tehnologic ce înlocuiește culegerea cu plumb pe linotip, monotip etc. cu culegerea pe hîrtii sau filme fototehnice prin intermediul unor automate de expu-



Schema 1 - Vechiul proces redacțional



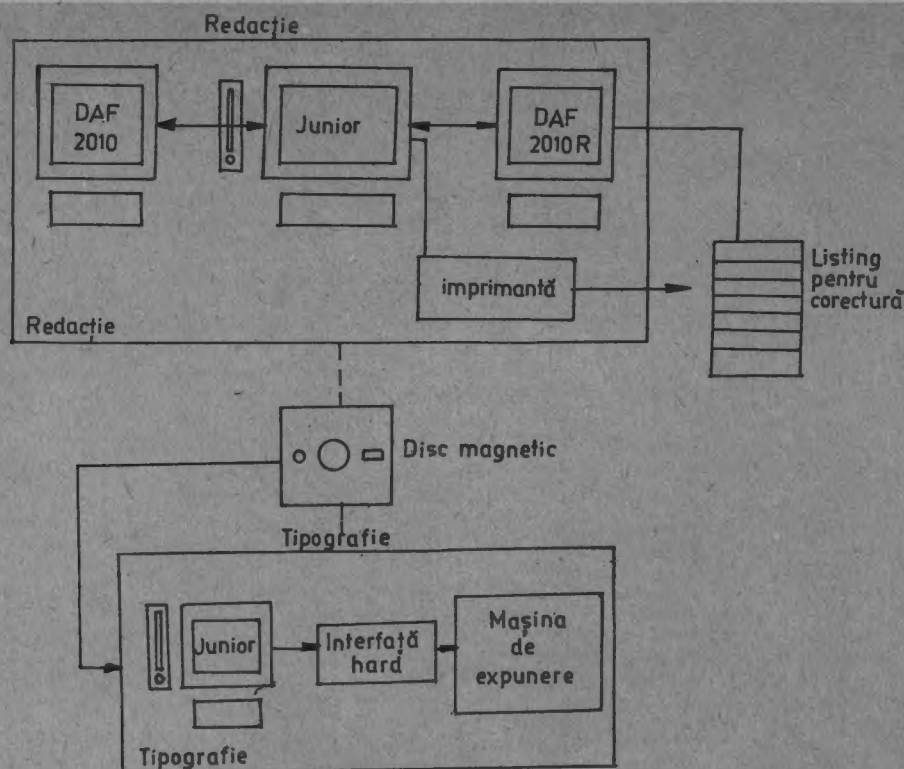


FIGURA 1

nere a textului. Practic, toate acestea au integrate calculatoare de proces specializate, iar productivitățile lor variază de la zeci de mii până la multe milioane de semne pe oră în funcție de generația căreia îi aparțin (și în fotoculegere specialiștii se referă la un număr de 4-5 generații, la fel ca în domeniul calculatoarelor de uz general).

## Calculatorul intervine

Comparativ cu prima schemă, activitatea redacțională asistată de calculator se prezintă ca în schema nr. 2. Este ușor de remarcat că nu mai sînt necesare înregistrările repetate ale textului condiționate de modificări multiple (intervenții, adăugiri și eliminări masive) și alte tipuri de corecturi, sau de prelucrarea textului pe mașinile de cules. De asemenea se observă că întregul proces de corectură (semnalarea și eliminarea erorilor) se face în interiorul redacției și

că tipografia nu mai poate influența, datorită schimbului de corecturi, termenele și prioritățile de apariție.

Sistemul redacțional-poligrafic ce funcționează în cadrul redacției revistei „Știință și tehnică” este bazat pe microcalculatoare pe 8 biți compatibile CP/M și este format exclusiv din componente de tehnică de calcul produse de IEPER-București.

În figura 1 sînt prezentate configurația acestui sistem, fluxul tehnologic al activității redacționale și legătura cu mașinile de expunere ale tipografiei Combinatului Poligrafic din Capitală. Componentele configurației au fost alese, din considerente tehnologice și de eficiență economică, din sortimentul de produse al întreprinderii sus amintite oferit la data elaborării programului. Locul de muncă al operatorului, corectorului sau redactorului la înregistrarea textului este videoterminalul DAF 2010R - varianta poligrafică.

Microcalculatoarele Junior sînt folosite

ca servere de fișiere pentru terminalele de culgere DAF 2010R. Fișierele pot fi atît programele de funcționare ale DAF-urilor, cît și articole sau fragmente de text. Imprimantele, cuplate și ele la aparatele Junior, produc listing-uri cu un set extins de caractere (circa 128), cu care se poate face culegerea de texte în limbile română, franceză, engleză, germană, italiană, spaniolă, maghiară; este identic cu setul implementat pe DAF-uri.

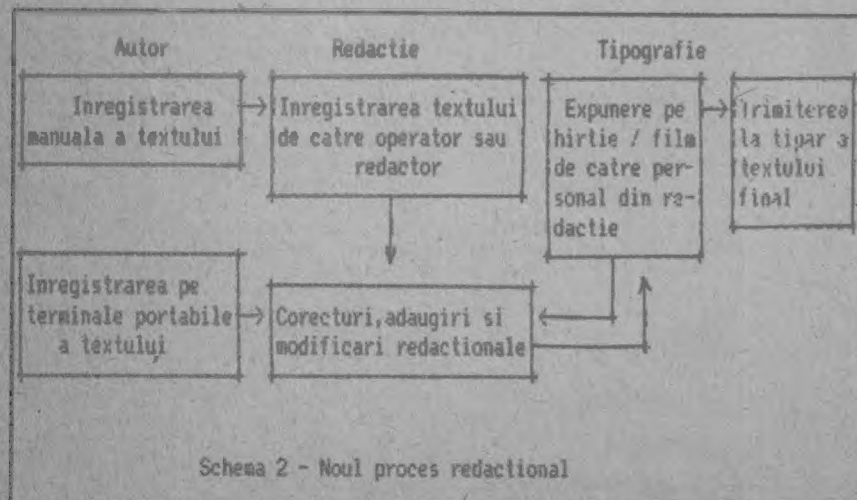
Un alt echipament Junior, plasat în tipografie, preia discurile cu textele finale obținute în redacție și le transmite mașinilor de expunere prin intermediul interfeței hard și al programului de transmisie SEND, elaborat de un specialist al IEPER - ing. Gabriel Dulcu.

Pachetul de programe de calcul pentru activitatea redacțional-poligrafică a fost elaborat de către autorul articolului de față. Două dintre acestea vor fi prezentate în continuare.

## Program dispecer în redacție

RIPALL este implementat pe Junior și asigură conducerea sistemului aflat în redacție. Un meniu permite alegerea unuia din cele două canale de legătură cu videoterminele DAF 2010R și conținutul transmisiei: a) transmiterea programului redacțional-poligrafic; b) recepția de texte înregistrate sau corectate și depunerea lor pe discuri flexibile și c) transmiterea textelor de pe discuri către DAF-uri, după o conversie prealabilă din codul TTS (folosit de mașinile de fotoculegere Harris 3300) în codul ASCII. Ultima operație poate fi însoțită și de transmiterea textului către imprimantă, sau această transmisie poate fi făcută numai către imprimantă.

Pe discuri, textele sînt înregistrate sub formă de fișiere în cod TTS și ele reprezintă articole întregi sau fragmente de articol.



Schema 2 - Noul proces redacțional

Aplicații pentru toți

## Program pentru activitatea redacțional-poligrafică

PLST este un program destinat realizării activității redacțional-poligrafice. În prezent, el este compus din următoarele module: înregistrare și editare de text, prelucrare poligrafică, culegere tabelară și culegere contorizată.

Modul de înregistrare și editare are câteva caracteristici mai deosebite ce merită menționate. Ecranul este împărțit în două zone: rîndul de stare afișat în negativ și zona de text afișată normal.

În timpul înregistrării, în rîndul de stare sînt actualizate, pe măsura scrierii în zona de text, valorile unor parametri poligrafici importanți ce trebuie să se afle în permanență în atenția operatorului, cum sînt formatul rîndului, familia, tăietura și corpul de literă, interlinierea, tipul de retrageri în text, iar în cazul culegerii tabelare și numărul coloanei de tabel în care se lucrează.

În rîndul de stare se semnalează, de asemenea, dacă se lucrează în regim de inserare de caractere sau nu; la o comandă se afișează fie numărul de caractere, fie numărul de rînduri și înălțimea (în puncte tipografice) ale textului final expus (în vederea machetării textului în pagină). Se afișează, de asemenea, în rîndul de stare numărul erorii comise la înregistrarea textului, precum și tipul de culegere specială în care se lucrează (culegere tabelară sau contorizată).

Lucrul din zona de text are și el câteva trăsături specifice. Să menționăm în primul rînd culegerea cu respectarea regulii cuvîntului întreg (wrap word), cunoscută și la alte programe editoare de texte. Textul nu este compus numai din semnele ce vor apărea în final pe hîrtie sau film, ci este însoțit de comenzi poligrafice plasate acolo unde trebuie să înceapă efectul lor. La înregistrarea textului, cînd se scrie o comandă poligrafică din cele afișate în rîndul de stare, valoarea nouă a acesteia este actualizată și în poziția comenzii în rîndul de stare. La înregistrarea textului singura comandă de corectură este ștergerea ultimelor caractere scrise. În general, adevărata corectură se face prin intrarea în regim de editare, care se obține automat prin deplasarea cursorului în interiorul textului. Sînt posibile operațiile elementare de corectură, caracter cu caracter, precum suprapunere de caracter, inserare sau eliminare de caracter. În ambele cazuri cu respectarea regulii de wrap word.

Operațiile de corectură mai complexă se efectuează pe blocuri de lungime variabilă. Una dintre acestea este anularea de blocuri de text. Prima comandă definește începutul blocului, iar cea de-a doua sfîrșitul, concomitent cu schimbarea afișării întregului bloc definit (de exemplu în video negativ). Întrucît există pericolul pierderii unor cantități mai mari de text prin manevrarea neatență, comanda de anulare nu poate fi executată decît după confirmarea ei prin repetare. La prima comandă de anulare afișarea blocului se face în mod clipitor, tip de afișare ce se constituie astfel într-o formă de avertizare sau alarmare ce nu poate fi trecută cu vederea.

O altă corectură complexă este inserarea de blocuri de cîte 256 de semne fără modificarea afișării la fiecare caracter. Se creează o zonă de trei rînduri de blankuri, în care se înregistrează cu mare rapiditate textul inserat. Repetarea comenzii înainte de epuizarea numărului de semne duce la îmbinarea sfîrșitului de bloc cu textul rămas, iar depășirea acestui număr conduce la inserarea de tip elementar, respectiv caracter cu caracter.

O a treia corectură complexă este muta-

rea de blocuri, cu sau fără anularea acestora în vechea poziție, și inserarea blocului în noua poziție. Despre corectura prin reîmplinire vom vorbi ceva mai încolo.

Textele pot fi afișate pe ecran în două moduri: neîmplinit sau împlinit. Textele împlinite sînt textele care formează o coloană în cadrul căreia lungimea rîndurilor de cuvinte este egală cu o anumită dimensiune (dată de tehnoredactor). Textele neîmplinite ocupă întreg ecranul. Modul neîmplinit este folosit de obicei la înregistrarea primară, cînd masivul de text nu a fost prelucrat conform comenzilor poligrafice care determină formarea unor rînduri ce corespund celor ce vor fi expuse pe materialele fototehnice.

După prelucrarea poligrafică textul poate fi afișat sub forma unor rînduri identice în conținut (nu și în lungime) cu rîndurile ce vor fi expuse pe mașina de fotoculegere. Eliminarea cîtorva caractere din aceste rînduri este permisă căci nu influențează aspectul rîndului expus. Inserarea de caractere cere de obicei corectura unor paragrafe (aliniate) prin reîmplinire. Pentru aceasta se efectuează corectura complexă cerută, se definește un bloc de text de mărimea unui paragraf sau unul care pornește de la începutul rîndului unde apare prima intervenție de corectură și se sfîrșește la capătul paragrafului. Corectura prin reîmplinire este foarte eficientă căci prelucurează numai fragmente reduse și nu întregi fișiere de text.

Tot în modul de înregistrare și editare se încadrează și tipurile de defilare a textului pe ecran (scrolling). Textul poate fi făcut să defileze rînd cu rînd sau pagină cu pagină, înainte și înapoi, prin repetarea comenzii sau poate fi afișat de la începutul sau sfîrșitul lui prin comenzi unice.

Modulul de prelucrare poligrafică are ca scop obținerea de rînduri împlinite (care prezintă pe ecran sau pe listing cu exactitate conținutul viitoarelor rînduri expuse). Pe ecran sau pe listing textul este însoțit și de comenzile poligrafice, în timp ce textul expus redă efectul lor.

## În ce constă prelucrarea poligrafică?

După cum s-a mai spus, fiecare literă sau semn de tipar are de obicei lățimi care diferă între ele în cadrul aceleiași tăieturi, lățimi care au alte valori dacă se schimbă tăietura, corpul sau familia de litere. Ele sînt exprimate în unități relative.

Prima comandă poligrafică luată în considerare este formatul rîndului, care, prin calcul, este transformat din cicero în puncte tipografice și apoi în unități relative. Fiecare semn sau literă al cărei cod a fost prelucrat capătă o lățime recalculată și în funcție de tăietura, corpul și familia de litere cu care se lucrează. Lățimile sînt însumate în cuvinte, cuvintele sînt însumate în valoarea curentă a rîndului, ce este comparată cu valoarea dată a formatului. Distanțele dintre cuvinte (spații, blankuri), luate cu valoarea lor minimă, sînt și ele însumate la valoarea curentă a rîndului. După fiecare însumare, în rîndul curent se adaugă produsul dintre numărul de blankuri și valoarea extensiei admise pentru blankuri (stabilită în funcție de criterii de estetică a culegerii). Dacă prin această ultimă însumare se depășește formatul rîndului, atunci rîndul se poate împlini numai prin extinderea blankurilor, dacă nu se adaugă încă un cuvînt, iar dacă prin însumarea acestuia se depășește formatul, atunci ultimul cuvînt este introdus în rutina de despărțire în silabe pentru limba română. Aici, din coada cuvintelor sînt eliminate silabe și înlocuite cu cratime. Operația se termină atunci

cînd partea de cuvînt rămas încapă în formatul rîndului. Se plasează la capătul rîndului caracterul de sfîrșit de rînd, silabele eliminate sînt înscrise în noul rînd curent, care se împlinește ca și precedentul, și tot așa pînă la sfîrșitul textului. După terminarea fiecărui rînd se face conversa codurilor de text și de comenzi poligrafice, care sînt de tip ASCII, în coduri de tip TTS pentru mașina de fotoculegere.

Textul în coduri TTS este transferat către Junior și depus ca fișier pe discuri magnetice flexibile prin intermediul programului RIPALL.

Readucerea textului în memoria DAF-ului în vederea modificărilor de efectuat de către corectori, redactori sau conducătorii redacțiilor se poate face sub două forme, condiționate de mulțimea corecturilor de efectuat. Cînd numărul corecturilor este mare, cînd acestea sînt voluminoase sau cînd se schimbă parametrii culegerii, se optează pentru forma de prezentare neîmplinită. Cînd este vorba de corecturi puține ale unor caractere răzlețe sau corecturi mai mari, localizate în unele paragrafe, se alege forma împlinită. Corecturile sînt urmate de înregistrarea unor noi fișiere de text. Ciclul de corectură continuă pînă la obținerea produsului final corectat care este pe film. Semnalările de corectură făcute pe listing-ul de imprimantă de către corectori, redactori sau conducătorii redacțiilor sînt efectuate pe fișierele de text primare afișate pe DAF. Listing-urile cu text împlinit, corectat sau final pot servi și ca material de arhivare în vederea urmăririi fazelor editoriale.

Modul de culegere tabelară permite culegerea unor table cu pînă la 50 coloane într-un rînd. La trecerea de la o coloană la alta se modifică în rîndul de stare parametrii coloanei. Pe măsura culegerii cîte unui semn într-o coloană, valoarea formatului coloanei se reduce cu lățimea semnului cules pînă la epuizarea formatului, cînd culegerea în continuare este blocată în mod automat. Afișarea în rîndul de stare a variației restului de format permite operatorului să ia deciziile corespunzătoare la culegerea și la trecerea de la o coloană la alta. La terminarea rîndului tabelar, format din mai multe coloane, parametrii poligrafici ai primei coloane sînt din nou afișați în rîndul de stare.

Modulul de culegere contorizată vizualizează și el în rîndul de stare variația formatului unui rînd și blochează culegerea rîndului la epuizarea formatului. Cînd mărimea acestuia se reduce sub un sfert din valoarea lui inițială, literele din rînd încep să fie afișate în modul stabilit, iar cînd se intră în zona de împlinire cu ajutorul extensiilor de blankuri la acest tip de avertizare se adaugă și cel sonor. Ieșirea din starea de blocare se face prin ștergerea consecutivă a ultimelor semne pînă la locul unde se poate plasa o cratimă. După introducerea acesteia și a caracterului de sfîrșit de rînd, semnele șterse din rîndul precedent sînt readuse în noul rînd, bineînțeles cu readucerea și afișarea formatului curent în rîndul de stare. De asemenea, în rîndul de stare este afișată în permanență informația că operatorul lucrează în acest mod de culegere (lucru valabil și la culegerea tabelară).

Modul de culegere contorizată este folosit pentru culegerea unor rînduri răzlețe, pentru corecturi finale, dar este util și la culegerea unor texte în limbi străine pentru care nu s-au elaborat programe de despărțire în silabe, deciziile de terminare a rîndului fiind luate de operator.

În încheiere trebuie arătată că sistemul poate fi aplicat, fără nici un fel de modificări, și în actuala secție de fotoculegere din Combinatul Poligrafic din Capitală.



# RUTINĂ GRAFICĂ pentru umplerea unor contururi

(urmare din pag.31)

**Utilizare:** rutina de umplere se apelează din BASIC de la adresa TEST, punctul de start fiind ultimul punct desenat cu PLOT sau PLOT INVERSE. Dacă se solicită umplere, punctul de start este marcat cu PLOT INVERSE 1, iar dacă se solicită ștergere, cu PLOT.

Următorul program BASIC exemplifică utilizarea rutinei (care se lansează de la adresa 63089), realizând umplerea și apoi ștergerea unui cerc de rază 30. Utilizatorul va introduce coordonatele centrului cercului sau va modifica programul, astfel încât să se realizeze umplerea sau ștergerea pentru alte contururi.

```

9 INPUT I
10 CIRCLE I,I,30
20 PLOT INVERSE 1:I+1
30 RANDOMIZE USR 63089
32 PAUSE 0
40 PLOT I+1,I+1
50 RANDOMIZE USR 63089
60 GO TO 9
    
```

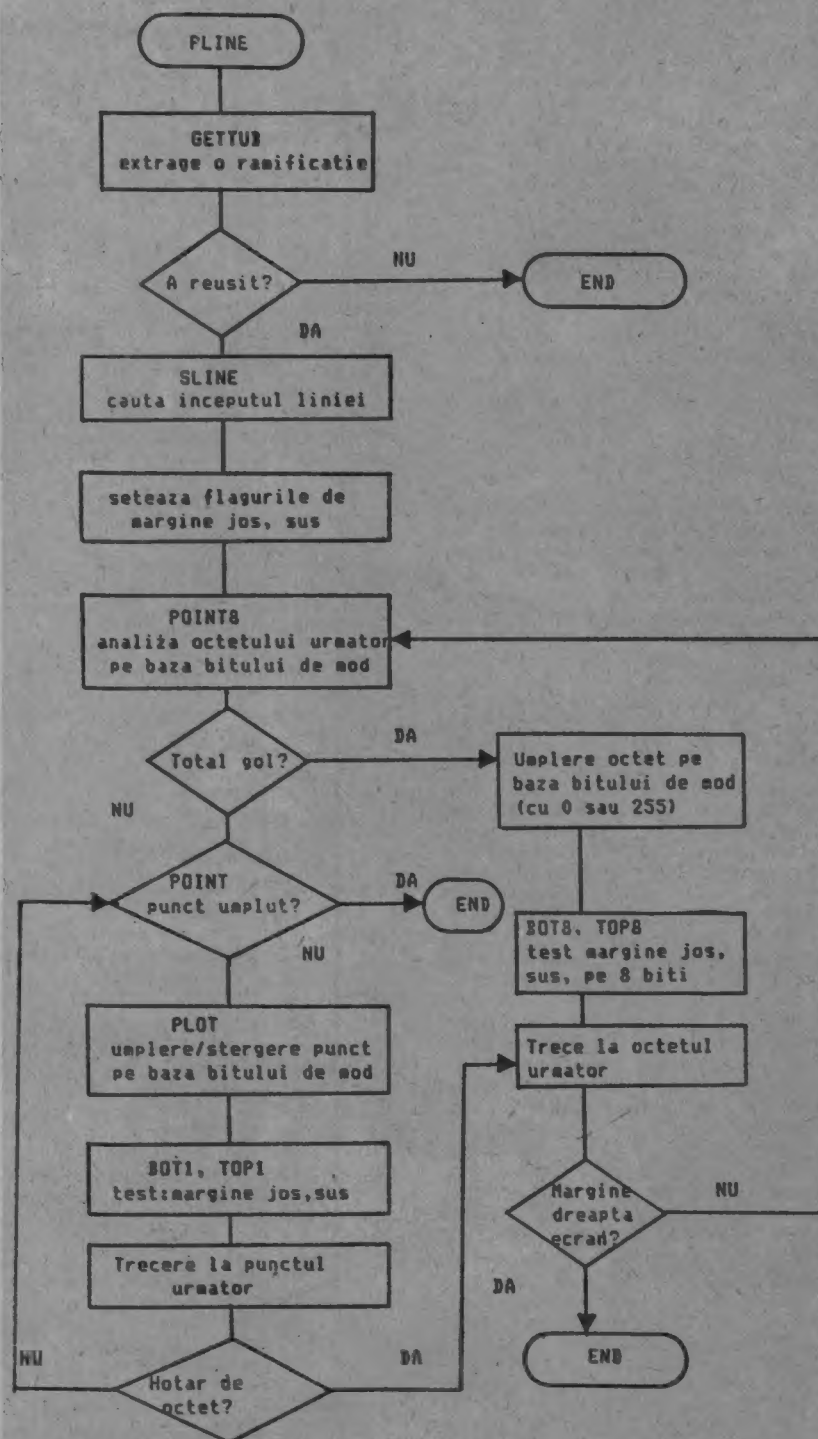
În vederea funcționării pe alte calculatoare care au memorie ecran astfel organizată sînt necesare mai multe modificări. De exemplu, pentru calculatorul PRAE 48 Ko (la care adresa de început a memoriei ecran este E000, iar cea de sfîrșit FFFF) sînt necesare următoarele modificări:

Adresă	Modificări	pentru
1000		
1001	memoria ecran	
ca la		
1002	PRAE	
1004		
1006	SCREEN:	
1008	EQU \$E000	
1010		
1012	SCREND:	
1014	EQU \$FFFF	
1016		
1018	PIXADD:	
1020	XOR A	
1022	SRA B	
1024	RR C	
1026	RRA	
1028	SRA B	
1030	RR C	
1032	RRA	
1034	SRA B	
1036	RR C	
1038	RLA	
1040	RLA	
1042	RLA	
1044	LD HL,SCREEN	
1046	ADD HL,BC	
1048	RET	
1050		
1052	HLVP:	
1054	PUSH BC	
1056	XOR A	
1058	LD BC,32	
1060	SBC HL,BC	
1062	POP BC	
1064	RET	
1066		

```

1068 HL,DOWN:
1070 PUSH BC
1072 LD BC,32
1074 ADD HL,BC
1076 POP BC
1078 RET
    
```

Rutina a fost experimentată de Szabo Attila și Toth Levente (Liceul Bolyai, Tg. Mureș) la tabăra de informatică pentru elevii de liceu de la Tirgu Mureș, 1989.



**Schema logică pentru rutina PLINE**  
(realizează umplerea propriu-zisă)

25 de milioane de oameni citesc,  
în fiecare lună, publicațiile noastre!

**MACWORLD**  
The Essential Magazine  
March 1991 \$2.95

**MAC VS P**  
Real they compare?

**Commodore 1990 RUN Index**  
**RUN**  
THE COMMODORE 64/128 USER'S GUIDE  
Tips For **PROGRAMMING SUCCESS**  
How to Sell Your Software  
Secrets to Factor Basic  
Try This Point 'N' Click Interface  
Address Envelopes

**1991 Design Contest Winners**  
**Publish**  
CD-ROMs For Print Publishing

**IDG**  
INTERNATIONAL DATA GROUP

● O rețea mondială care acoperă  
90% din piața calculatoarelor!

● Liderul mondial al publicațiilor  
de informatică!

**NETWORK WORLD**  
The Newsworld of User Networking Strategies  
Volume 8, Number 2  
Desert Storm puts pressure on U.S. acts  
Satellites help U.S. forces control the skies over Iraq  
Tools for distribute  
32-Bit Way O

**COMPUTERWORLD**  
FORECAST 1991  
Blah Busters!

**Digital NEWS**  
DEC 20 results, layoffs  
Storage Concepts RAID 17  
20/20 v.3.0 for VMS  
DECUS hacker dispute  
Token Ring an obstacle to full net access  
Storage must be managed

**FEDERAL COMPUTER WEEK**  
THE NEWS WEEKLY FOR THE  
Cockpit Maps Help Pilot Gulf Sorties

**The Adventures of GAMEMASTER**  
The 2nd Game Dimension!

**CIO**  
COVER STORY:  
BE ENGINEERING  
BUSINESS  
PROCESSES

**Workstation**  
Buyer's Guide: 41 PORTABLE NOTEBOOK COMPUTERS... PAGE 28  
**PORTABLE OFFICE**  
Cellular Across America

Orice informație de la dumneavoastră este,  
prin intermediul revistei INFOCLUB, pentru întreaga lume!